1988

A proposal for a microcomputer based system to automate the Marine Corps Crime Statistics Reporting Program

Paquette, Paul Emile.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

A PROPOSAL FOR A MICROCOMPUTER BASED SYSTEM TO AUTOMATE THE MARINE CORPS CRIME STATISTICS REPORTING PROGRAM

by

Paul E. Paquette

March 1988

Thesis Advisor Barry A. Frew

UNCLASSIFIED

Security Classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification: Unclassified	1b Restrictive Markings
2a Security Classification Authority	3 Distribution Availability of Report: Approved for public release; distribution is unlimited.
2b Declassification/Downgrading Schedule	
4 Performing Organization Report Number(s)	5 Monitoring Organization Report Number(s)
6a Name of Performing Organization: Naval Postgraduate School; 6b Office Symbol (*If Applicable*): 37	7a Name of Monitoring Organization: Naval Postgraduate School
6c Address (*city, state, and ZIP code*): Monterey, CA 93943-5000	7b Address (*city, state, and ZIP code*): Monterey, CA 93943-5000
8a Name of Funding/Sponsoring Organization; 8b Office Symbol (*If Applicable*)	9 Procurement Instrument Identification Number
8c Address (*city, state, and ZIP code*)	10 Source of Funding Numbers: Program Element Number; Project No; Task No; Work Unit Accession No

11 Title (*Include Security Classification*) A PROPOSAL FOR A MICROCOMPUTER BASED SYSTEM TO AUTOMATE THE MARINE CORPS CRIME STATISTICS REPORTING PROGRAM

12 Personal Author(s) Paquette, Paul, E.

13a Type of Report	13b Time Covered	14 Date of Report (*year, month,day*)	15 Page Count
Master's Thesis	From To	1988 March	197

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms (*continue on reverse if necessary and identify by block number*)
Field	Group	Subgroup	Database Management Systems, Structured Design, Systems Analysis

19 Abstract (*continue on reverse if necessary and identify by block number*)

This thesis investigates the possibility of implementing a Database Management System to support information processing needs within the Military Police Section of the Plans, Policies and Operations Department of Headquarters Marine Corps. An overview of the structured analysis and design methodologies with emphasis on the Life Cycle approach to software engineering was conducted. The numerous tools provided by the structured analysis and design methodologies were utilized in the development of the Database Management System. This implementation emphasized documentation and maintenance to ensure that the system will meet the current and future needs of the Military Police Section.

20 Distribution/Availability of Abstract	21 Abstract Security Classification	
[X] unclassified/unlimited [ ] same as report [ ] DTIC users	Unclassified	
22a Name of Responsible Individual: LCDR Barry A. Frew	22b Telephone (*Include Area code*): (408) 646-2772	22c Office Symbol: 54Fw

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted

All other editions are obsolete

security classification of this page

UNCLASSIFIED

Approved for public release; distribution is unlimited.

A Proposal for a Microcomputer Based System to Automate the Marine Corps Crime Statistics Reporting Program

by

Paul Emile Paquette
Captain United States Marine Corps
B.A., University of Maine, 1979

Submitted in partial fulfillment of the requirements for
the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March 1988

ABSTRACT

This thesis investigates the possibility of implementing a Database Management System to support information processing needs within the Military Police Section of the Plans, Policies and Operations Department of Headquarters Marine Corps. An overview of the structured analysis and design methodologies with emphasis on the Life Cycle approach to software engineering was conducted. The numerous tools provided by the structured analysis and design methodologies were utilized in the development of the Database Management System. This implementation emphasized documentation and maintenance to ensure that the system will meet the current and future needs of the Military Police Section.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to thank the following people without whose help much of this thesis would not have been possible.

First I would like to thank my wife Joyce and my son Aaron. Their many words of encouragement and support made this thesis possible.

Secondly I would like to thank my thesis advisor, Barry Frew and my second reader, Professor Ben Roberts. I could not have been instructed by two finer individuals. To them I owe a debt of thanks.

I. INTRODUCTION

An interesting analogy can be drawn between the industrial revolution of the 18th century and the "information revolution" of today. In much the same manner as the industrial revolution attempted to replace muscle with machines to increase worker productivity, todays information revolution uses the strength of computers to enhance worker productivity.

The need to be better able to manage information in todays business settings cannot be understated. There exists an ever increasing demand to administer large quantities of data for a wide variety of commercial, governmental and administrative purposes. It has been estimated that the typical executive spends 80 percent of his time in the processing and communication of information [Ref. 1:p. 4]. Any technology which would allow him to accomplish these tasks faster, and with fewer revisions would certainly be beneficial.

Computers have become an integral part of an organization's information processing scheme for two important reasons. The first reason is due simply to the sheer volume of data with which an executive must contend. The second reason is due to the advances in computer technology. The past several years have seen quantum leaps in microcomputer and storage technologies. This fact has caused the cost of obtaining these technologies to plummet. The capabilities of just a few short years ago, which were found only in expensive mainframe computers, are now available to virtually any business concern.

As the information processing needs of businesses became more complex, and the business environment became more competitive, the view of data as a valuable resource began to emerge. This view of data as a resource gave rise to various

strategies for the management of information resources. The need for database in the management of information was obvious. Data was simply the raw material for producing items of information. In order to be available for retrieval, processing and use, the data had to be managed by some sort of Database Management System.

Database Management Systems were initially developed in the late 1960's for mainframe computers. Their main function was to collect, store, organize and output a collection of related data. They sought to avoid data duplication, facilitate coordination and maintain the integrity of the data. With the advent of the microcomputer and Winchester disk technology, the concept of a Database Management System could easily be implemented in many more settings. The purpose of this thesis was to explore various design and implementation strategies for a Database Management System in the microcomputer environment.

A. BACKGROUND

The Operations Division, Plans, Policies, and Operations Department, Headquarters Marine Corps, requested that a student at the Naval Postgraduate School investigate the possibility of designing and implementing a Database Management System to support information processing needs within the Operations Division. Two primary areas of concern were the maintenance of crime statistics for the Marine Corps and the management of the Physical Security Waiver and Exception Program. Both programs involve the tracking of requests, filing of reports, and responding to ad hoc queries. Initially the system was entirely manual. Clerks sifted through reams of documents in order to respond to ad hoc queries. The production of monthly and annual reports required a great deal of clerical support and response time to requests for information was

excessive. This research was conducted in order to recommend and implement a Database Management System which would meet the current and future needs of the Operations Division.

1. Objective

The objective of this thesis was to provide a system which was tailored to the needs of the Military Police Section of the Plans, Policies and Operations Department of Headquarters Marine Corps. The system should decrease the man hours required to implement the Crime Statistics Reporting Program and allow the Military Police Section to respond to ad hoc queries in a timely and efficient manner. In so far as possible the system was designed to be maintainable, reliable and adaptable.

2. Research Questions

The research questions were:

(1) What is an appropriate database management system to recommend for use at a location which enjoys limited hardware and clerical support?

(2) Which database implementation best satisfies the user's needs in accordance with specified data manipulation requirements?

(3) Is the current information generated by the system adequate? Are ad-hoc requests adequately supported?

(4) What will be the retrieval, update, and security requirements of the Database Management System?

(5) How can we assure ease of use?

3. Scope

The Military Police section is charged with administering a wide range of programs which encompass many different reporting requirements. The production of the annual Crime Statistics Report, the administration of the Physical

Security Waiver and Exception Program and the Dependent Child Identification Program are just a few of the programs which are administered by the Military Police section.

Although all of the programs could be enhanced by some form of automated update and retrieval system, the emphasis of this research was on developing a system which would automate the Crime Statistics Reporting requirements. The focus was on this program because it was, by far, the largest program in terms of reporting requirements, and it could be used to establish the foundation for a system which could be expanded to incorporate other programs.

The cost of research and implementation of a Database Management System for the Military Police Section is not expected to exceed $2,000.00. This sum includes travel expenses to Washington D. C. , the purchase of an application program and the necessary hardware.

4. Methodology

The conceptual foundation within which the questions will be examined is that of the structured design methodologies. A thorough literary study of the methodologies as proposed by Yourdon, Constantine, Page-Jones et al was conducted. Emphasis was placed on a life cycle approach to systems design, software planning, requirements, analysis and structured coding. To this end, a thorough analysis of the existing system was undertaken. After relevant specifications such as costs, benefits, schedules, and performance requirements were established, structured design begun. Data Flow Diagrams, Data Dictionaries, Structure Charts etc. were utilized in the design phase. Once the design was completed, a prototype of the project was coded. The prototype was intended to be a shell of the complete project which allowed the user to interact

with the proposed system. Various screens were presented to the user in an effort to ensure design requirements and to identify potential problems. After the prototype had been verified, the coding began. This structured implementation was done with emphasis on documentation and maintenance considerations.

5. Summary of Findings

The automation of the Marine Corps Crime Statistics Reporting Program had an impact in two distinct areas. The first of which was the timeliness and accuracy of the reporting requirements and the second dealt with the nature of the data which was collected to generate the Annual Crime Statistics Report.

Because the crime statistics for individual installations are now stored in a database, vice a manual filing system, ad hoc queries and normal reports can be done more efficiently. Before the database was established, the production of the annual Crime Statistics Report was an effort which spanned several weeks. It now takes only several minutes to produce the same report, and of course there are no arithmetic errors during the computation of the figures.

The second finding dealt primarily with the nature of the data used to generate the annual report. As was discussed during the data modeling section of this thesis, a majority of the data collected via NAVMC 1630/1 is not very useful to the annual report. As it is presently configured NAVMC 1630/1 collects a good deal of information about specific crimes, attributes of those crimes (i.e on base, off base, investigated by NIS etc.) and the status of the perpetrator (USMC, NAVY, Civilian, Dependant, etc.). It does not, however, allow us to relate this information in a truly meaningful way. Nor is this information (crime attributes and status of perpetrator/victim) necessary to the Annual Report. This finding will be elaborated upon in the recommendations section of the thesis.

II. STRUCTURED ANALYSIS

The process of developing a system can be seen as comprising seven different phases: problem definition, feasibility study, analysis, system design, detailed design, implementation and maintenance. Although presented in a linear fashion, it should be remembered that there will be a good deal of interaction and feedback between the phases of the System Life Cycle. As the system evolves from conceptualization to implementation, each one of the phases of the System Life Cycle will have been visited. When a structured approach is used, the systems analyst must progress from step to step in a careful, methodical fashion, completing a number of well defined exit criteria for each step [Ref. 2:p. 9].

The primary objective of the analysis phase is to study the problem prior to taking action. The focal point of analysis in the very early stages is problem definition. That this is the starting point of the system life cycle would seem to be common sense but as Davis noted ..."although the need for problem definition may seem obvious, this is perhaps the most frequently bypassed step in the entire systems analysis and design process." [Ref. 2:p. 9]

A. PROBLEM DEFINITION

The problem definition begins to take form with the realization by the user that the present way of doing business does not adequately meet his needs. This could be due to a host of factors such as the expense of the current system, the ability of the system to respond to ad-hoc queries and the level of skill required to use the system etc. In the case of the Military Police Section of the Operations Division of Headquarters Marine Corps, the problem was to provide the Military Police

Section with a system which would efficiently compile, update and retrieve crime statistics as well as information on their Waiver and Exception program. The system, as it was initially configured, was entirely manual. A small clerical staff was responsible for the consolidation of monthly reports received from installation Provost Marshal Offices throughout the Marine Corps. The information from some 60 reports per month was used to generate an annual crime statistics report as well as respond to ad hoc queries requested throughout the reporting period. The annual report required that the data be presented in many different views. Indices were required for data attributes such as violent crimes, crimes against property, crime rate by base, and percentage change between reporting periods to name a few examples. The costs of generating this report, in terms of clerical support and time, were becoming exorbitant. The Military Police Section did not possess sufficient resources to produce this report in an efficient manner. As Meilir Page-Jones observed, "Typically, the user is concerned less about the way in which he's doing business than about the physical deficiencies of the system that he uses to do it . " [Ref. 3:p. 23] This certainly was the situation found in the Military Police Section.

Based on interviews with the Marines in the Military Police Section we were able to gain an appreciation of the problem faced by them. Our understanding of the problem was formalized in the problem definition phase of the Life Cycle Model. The intent of the formal problem definition was to define the scope and objectives of the proposed system. The Statement of Scope and Objectives served three important purposes:

(1) It assured that the analyst and the user viewed the problem in the same light.
(2) It gave the user a gross estimate of project costs

(3) It ensured that the analyst and the user agreed upon the general direction of the project.

The statement of Scope and Objectives as presented to the user is shown in Figure 2.1.

As noted by Davis, "The function of the Statement of Scope and Objectives is communication, it is a formal way of saying here is what I think you want. [Ref. 3:p. 24] The Statement of Scope and Objectives thus served as a tool which allowed the analyst and the users in the Military Police section to clear up any misunderstandings while the project was still in its infancy. After all of the parties had agreed that the Statement of Scope and Objectives accurately reflected their understanding of the problem, the next phase of the System Life Cycle had begun.

STATEMENT OF SCOPE AND OBJECTIVES	
THE PROJECT	Crime Statistics Reporting Program
THE PROBLEM	The Military Police Section is not able to adequately respond to ad hoc queries vis a vis their Crime Statistics and Exception Programs
PROJECT OBJECTIVES	To reduce the amount of time required to process monthly reports. To allow for statistical analysis of the data. To reduce the time and effort required necessary to respond to ad hoc queries.
PROJECT SCOPE	The development costs of this project should not exceed $2,000.
PRELIMINARY IDEAS	One possible solution would be to generate a database system on existing hardware
THE FEASIBILITY STUDY	In order to more fully justify the potential of this project, a feasibility study lasting approximately three weeks is suggested. The cost of this study should not exceed $500.00

Figure 2.1 Statement of Scope and Objectives

B. THE FEASIBILITY STUDY

According to Davis, "A feasibility study is a high level capsule version of the entire systems analysis and design process. The objective is to determine quickly and at minimum cost if the problem can be solved." [Ref. 2:p. 30] The feasibility study utilizes the exit criteria of the problem definition phase of the System Life Cycle as input. In the feasibility study the problem definition is refined and the scope and objectives are clarified. The purpose of the feasibility study is to determine if the problem is worth solving. In order to assess whether a problem is worth solving, three separate facets of feasibility must be addressed:

(1) Technical: Can the system be implemented using current technology
(2) Economical: Do the benefits outweigh the costs?
(3) Operational: Can the system be implemented in this organization? [Ref. 2:p 274]

The first task of the analyst during the feasibility study was to attempt to understand the existing system. The existing system served as an important source of information. It gave an indication as to what basic functions needed to be incorporated into any new system as well as highlight those problems which we wished to avoid. It also served to bound the scope of the project. Obviously if the benefits realized by the new system did not outweigh the costs incurred, the new system would not be implemented.

In order to properly analyze and understand the existing system, the analyst had to learn the specifics of the Marine Corps Crime Statistics Reporting Program. Several sources of information were consulted. Marine Corps installations are primarily guided in their crime prevention programs by Marine Corps Order 1600.16a. Local orders, desk top procedures and standard operating procedures

are examples of other types of internal documentation which were consulted. The crime statistics reports which were produced in previous years also provided an insight to the systems functioning and requirements. The most valuable information however, was that which was gleaned from interviews with key people in the Military Police section. The functioning of the "informal system" was studied by interviewing personnel within the Military Police section as well as tracking the flow of work in the organization. This process is graphically illustrated in Figure 2.2. As can be seen from the illustration, the system was entirely manual. Twenty separate Provost Marshall Offices forwarded three monthly reports to the Military Police Section. The reports were filed by the personnel within the section. Whenever an ad hoc request was received by the Military Police Section, the requested information was retrieved from the files, compiled into a report format and then sent to the requestor. This process was tedious as well as time consuming. As an example, it was estimated that a simple query such as "How many homicides involving United States Marine Corps personnel were recorded in 1986?" involved viewing and tabulating 240 separate reports. The response time for this query was typically a week or more.

After numerous conversations and exchanges of correspondence with the Non-Commissioned Officer in charge, two primary areas of concern were noted. These were the maintenance of crime statistics for the Marine Corps and the inability of the Military Police Section to respond to ad hoc requests in a timely manner. It was felt that an automated system of update and retrieval would greatly enhance the services which the Military Police Section could provide to requesting organizations. In addition, a larger database could be more effectively manipulated such that historical trends could be analyzed. This information could be used to

assist commanders who identify crime trends and implement crime prevention measures that address the specific criminal activity in his area. Because the system was entirely manual, the production of ad hoc or annual reports required a great deal of time and clerical support. Any attempt at an historical analysis of the accumulated data was virtually intractable.

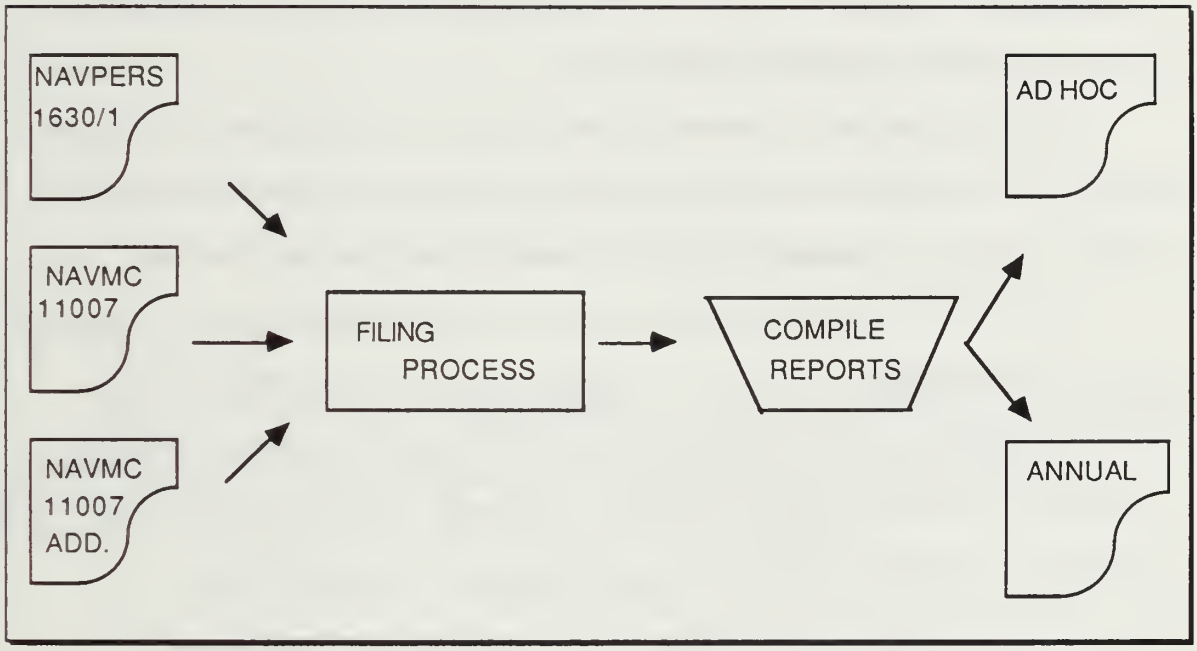

Figure 2.2 System Flow Chart

The feasibility study addressed three possible solutions to the problem as addressed in the Statement of Scope and Objectives:

(1) Get more clerical support

(2) Generate a database and implement a Database Management System on existing microcomputers

(3) Establish a link to the mainframe located at Marine Corps Base, Quantico, Virginia

The first proposal was not considered feasible due to the fact that the staffing requirements as set forth by Headquarters, Marine Corps, showed this unit to be at full strength.

The second option had no real drawbacks. It was certainly economically feasible (the hardware had already been bought), technically feasible and it was politically desirable (the Officer in Charge did not want to depend on anyone for the operation of his eventual system).

The third option was discussed at some length but it was felt that the cost of linking to the mainframe, training the clerks, and getting support from the data center at Quantico (definitely a political consideration) was not feasible.

The feasibility study also addressed the selection and acquisition of a potential Database Management System. The criteria which would be used to make this choice included economic and political feasibilities as well as administrative and technical criteria as proposed by Everest in his book, "Database Management" [Ref. 4:p 687]. The major points which would have to be considered were:

(1) Technical Criteria
    database definition
    generalized retrieval capabilities
    generalized update capabilities
    programmer interface
    database integrity
    modes of operation
    database evolvability

(2) Administrative Criteria
    product stability
    maintenance support

documentation
supporting environment
costs

Armed with the information which was gathered through a combination of interviews and reading applicable Marine Corps Orders, we were able to generate a representation of the physical system. The functioning of the Crime Statistics Reporting System as it existed in the Military Police Section is represented in the system flow chart of Figure 2.2. The systems flow chart is an excellent way to graphically portray the way the system functions. Data Flow Diagrams can be used to show the flow of data through the functions of the system.

C. ANALYSIS

The most obvious aspect of the system flow chart is that there are no automated representations. There is nothing indicating on-line storage, video display terminals or automated I/O. The system is entirely manual.

As was previously noted in the feasibility study, this system flow chart was used to gain an insight into how the Military Police Section produced any required reports. It is an excellent way to graphically portray the way the system functions in a physical, high level diagram. It does not, however, give us an insight into those functions which are essential to the Military Police Section.

1. Data Flow Diagrams

A fundamental tenant of Structured Analysis is that Data Flow Diagrams ..." be used to sweep aside as many irrelevancies as possible in how a user currently happens to do his business and uncover the logical functions essential to the way in which he has to do business." [Ref. 3:p. 70]

The Data Flow Diagrams represent a logical view of the system which is removed from any implementation considerations. As explained by DeMarco:

> The building blocks of a Data Flow Diagram are sources, destinations, processes, data stores and data flows. A leveled set of Data Flow Diagrams is made up of a top, a bottom, and a middle. The top is a single diagram called the context diagram. The bottom consists of a set of partitioned bubbles, called functional primitives, the middle is everything else. [Ref. 5:p. 75]

Data Flows are represented by arrows indicating which way data is flowing in our system. The label of the data flow is written alongside the arrow. All data names that are shown on the Data Flow Diagram are organized in a collection of logical definitions called a Data Dictionary.

Data sources are simply the place where the data item originated. They are represented by boxes on the Data Flow Diagram with data flows originating from them.

Data destinations are the places where the data is flowing toward. They are often referred to as data sinks but in any case they represent a receiver of data flows from the system. Note that a single box could represent both a source and a sink.

The different processes found in our system are represented as bubbles on the Data Flow Diagrams. DeMarco explains their purpose thusly:

> Processes are represented as bubbles on the Data Flow Diagram. A process is some action which serves to transform the data. This transformation can occur in either of two ways. Processes can transform the structure of the data, for example by reformatting it or they can transform the information contained in the data, for example changing a regular price to a trade-discount price [Ref. 3:p. 61].

Data stores are repositories of data. They are files in the sense that they hold information, but it is important to note that data sources do not represent any physical storage medium such as disk or tape. They are the logical place where data is stored in the system. Data stores are represented by a pair of parallel lines on the Data Flow Diagram.

To help us fully understand the various components which appear on the Data Flow Diagram, two more tools of Structured Analysis will be introduced. They are the Data Dictionary and the Minispec.

2. Data Dictionary

Once again we cite DeMarco for an explanation of another of the tools of structured design:

> The Data Dictionary contains definitions of all data mentioned in the Data Flow Diagram, in a process specification, or in the Data Dictionary itself. Composite data, (data that can be further divided) is defined in terms of its components; elementary data (data which cannot be divided) is defined in terms of the meaning of each of the values that it can assume. Thus the Data Dictionary is composed of definitions of data flows, data files, data used within processes and elementary parts of the above. [Ref. 3:p. 75]

An entry from our Data Dictionary, which is found in Appendix B, would look like this:

NAVMC 11007 = INSTALLATION NAME + INSTALLATION POPULATION + PERIOD + DUI ON BASE + DUI OFF BASE + DWI ON BASE + DWI OFF BASE

In other words, the data flow called NAVMC 11007 consists precisely of the items Installation Name, Installation Population, Period, DUI on base, DUI off

base DWI on base, DWI off base, concatenated together. All of these items must be present, and they must be in this order. No other data flow could pass as NAVMC 11007, even though the name might be applicable. Each element of the definition will also be rigorously described in elemental terms. For instance, Period was a component of the NAVMC 11007 definition which can be further partitioned into more fundamental elements:

Period = year + month

This top-down partitioning of our data will be done to every component shown on our Data Flow Diagram and for all subordinates used to define them. This partitioning will help to ensure that the data is used consistently and without ambiguity.

3. Minispecs

The method that we use to describe the data transformations, which occur in the lowest level bubbles of our Data Flow Diagram, is a structured English specification. This type of process description is also called a Minispec. The intent is to provide a very concise, rigidly structured statement of what it takes to do the data transformation. The minispecs use a ..."pruned down version of English embedded in the simple constructs of Structured Programming." [Ref. 3:p. 80] An example of a minispec written in structured English follows. The number 1.0 relates the minispec to the bubble that it functionally describes.

Process 1.0
FOR NAVPERS 1630/1 DO
    FOR NAVMC 11007 DO
        FOR NAVMC 11007 ADDENDUM DO

```
                    DO WHILE NOT EOF INSTALLATION FILE
                    LOCATE EACH INSTALLATION
                    UPDATE CRIME STATISTICS
                    ENDDO WHILE NOT EOF
                ENDDO
            ENDDO
        ENDDO
```

4. Context Diagram

The Context Diagram identifies the scope of analysis and specifies the systems inputs and outputs [Ref. 3:p. 67]. It is used to represent the system as a single logical process and serves to identify the sources and destinations of data. As such, it represents the highest possible view of the system.

Figure 2.3 is the Context Diagram of the Marine Corps Crime Statistics Reporting system. It is used to show the system in context to the environment in which it operates.

The Context Diagram is used primarily as a communication tool allowing verification of the data sources and destinations. As illustrated in Figure 2.3 the "source" of data is the various Marine Corps installations throughout the world. The data, as represented by NAVMC 11007, NAVMC 11007 Addendum and NAVPEPS 1630/1 are transformed by the Crime Statistics Reporting process. The ultimate destination of the now transformed data, the Annual Crime Statistics Report for example, is shown to be Headquarters, United States Marine Corps as well as the reporting installations of the Marine Corps.

Figure 2.3 highlights the high level functioning of the Crime Statistics Reporting Program. However, in order to fully understand the system, this

process must be decomposed into its functional parts. Referring back to the discussion in the Problem Definition, we were able to identify two constituent processes of the Crime Statistics Reporting process. They were: Update Files and Generate Reports. These two processes represent the basic functions that must be performed by the system. They are the bubbles which represent the lower level components of the context diagram bubble.

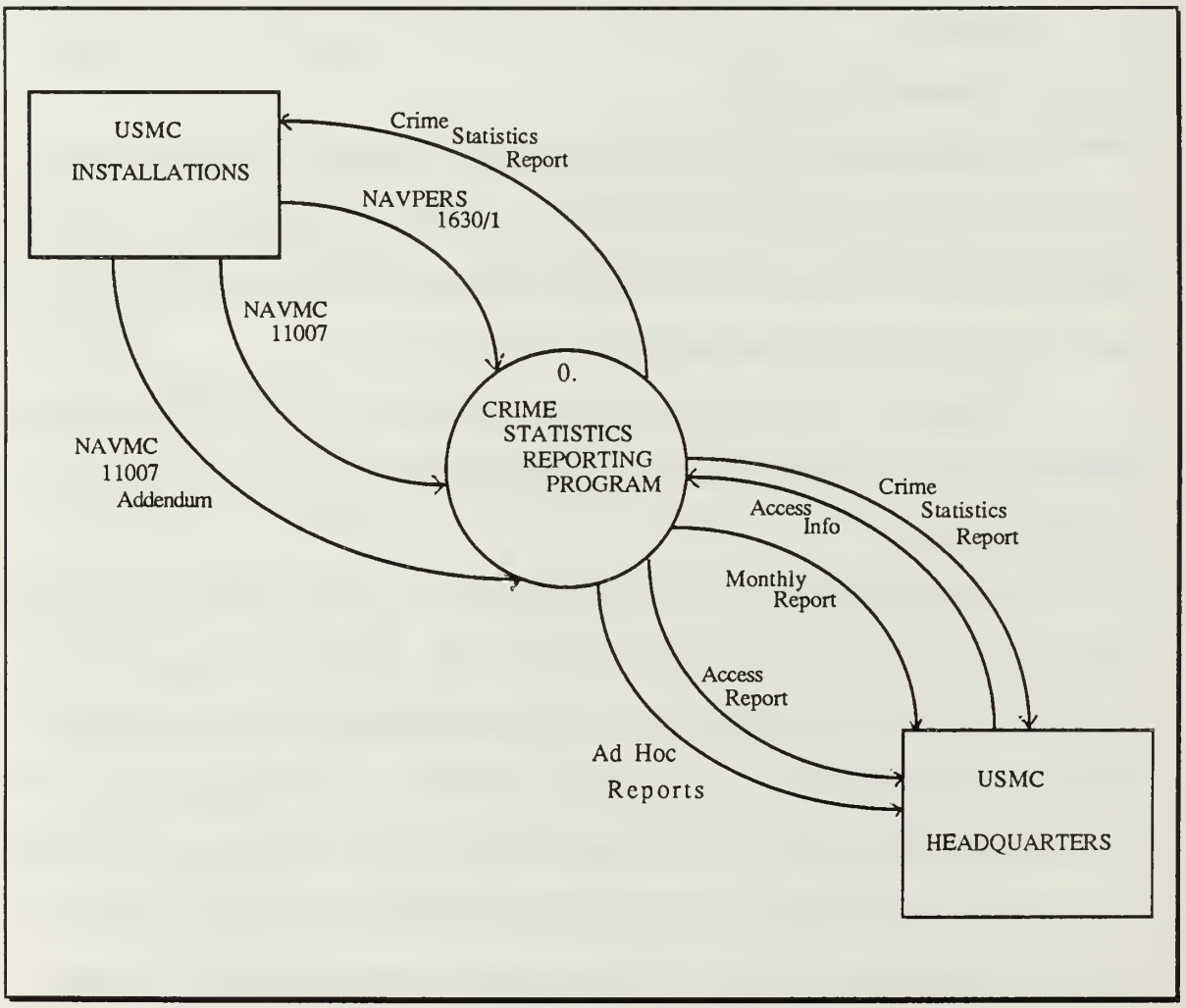

Figure 2.3 Context Diagram

After just a few decompositions our Data Flow Diagram becomes too large to fit on a single page. In order to keep the diagram meaningful DeMarco makes the following recommendation:

> When a system is too large for its Data Flow Diagram to be shown on a single page, we ought to do an initial partitioning into subsystems. If the subsystems are still too large, we will divide them into subsystems. And so on. Eventually we will end up with components that can be portrayed with simple Data Flow Diagrams of primitive functions. [Ref. 5:p. 72]

DeMarco recommends that we continue this leveling or partitioning of our system until the operation of a bubble can be described in about a page [Ref. 3:p. 66].

Figure 2.4 represents the first functional partitioning. It demonstrates how this leveling process works to clarify the functioning of the system. The context level process was decomposed into four sub processes, Security, labeled 1.0, Log-In, labeled 2.0, Update Files labeled 3.0, and Generate Specified Report, labeled 4.0. The dashed line around bubbles 1.0 and 2.0 represents bubble 0 and is intended only to clarify this leveling concept.

Note that the level one Data Flow Diagram shown in Figure 2.4 contains the same net inputs and outputs as does the Context Diagram. This fact demonstrates that the level one Data Flow Diagram is just a more detailed representation of the Context Diagram. Note also that the level one Data Flow Diagram shows several "new" data flows and four "new" files. These were not really new but rather existed as details of the Context Diagram that were not shown at that level. Now that the details of the Context Diagram have been exposed in the level one Data Flow Diagram, each component of the level one Data Flow Diagram

can be examined and further decomposed into its own constituent bubbles (processes) if necessary. This decomposition allows validation of the upper layers by producing and matching of the lower layers.

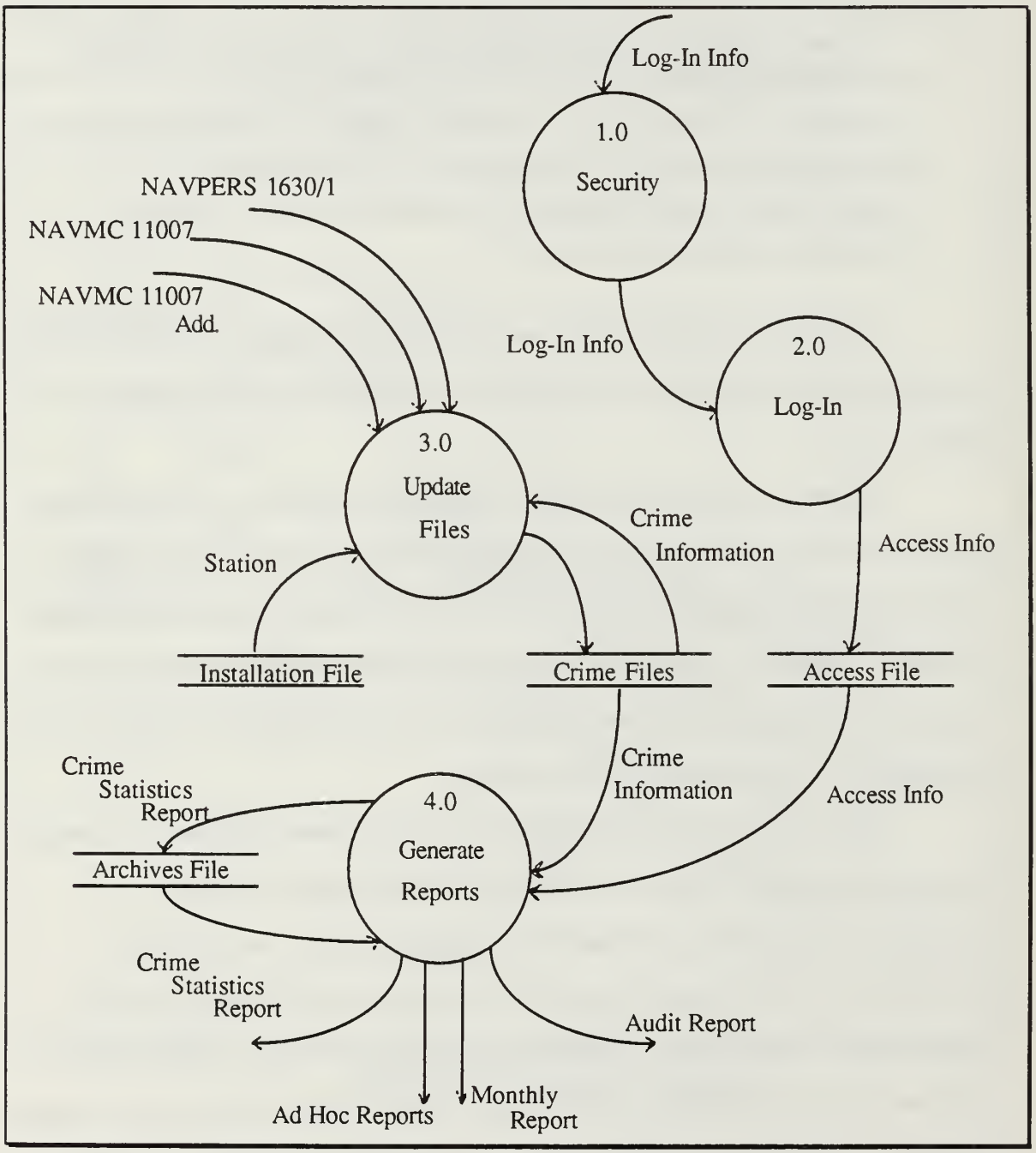

Figure 2.4 Level One Data Flow Diagram

5. Level One Data Flow Diagram

What follows is a walk through of Figure 2.4 and the rest of the leveled Data Flow Diagram set, conducted along the analysis phase walk through for the user. We begin with a quick overview of the top level before turning to the specific details.

The data flows which enter the system are represented by NAVMC 11007, NAVMC 1107 Addendum and NAVPERS 1630/1, all of which originate with the various Marine Corps installations. The data flow labeled "Log-in Info" originates with the user of the system. It is composed of a password, as well as the user's name, rank and unit.

The first process shown on the level one Data Flow Diagram is the security function labeled 1.0. The purpose of this function is to allow only those individuals who have the proper authorization to access the system. If an authorized user enters the correct password, he is allowed access to the other processes of the system. If an incorrect password is offered more than three times, the system will shut down preventing any operations on the system. Because its functioning can be described in about one page, the security process can be considered to be a functional primitive. [Ref. 5:p. 72] As such, a minispec was prepared for it and can be found in Appendix B.

The next process found on the level one Data Flow Diagram is "Log-In", process 2.0. After the user has successfully logged onto the system the Log-In process will take the information which is contained in the Log-In Info data flow and combine it with the date /time of access. It will also record the operations performed by the system (i.e.,update, delete, edit, reports). This information is written to the file labeled "Access Info". The process provides an audit trail of

accesses to the system. This process represents a functional primitive and its minispec can be found in Appendix B.

The third process shown on the level one data flow diagram is process 3.0, labeled Update files. The purpose of this process is to receive incoming information, alter the information into the form of data labeled "Crime Information" and store it in the appropriate Crime file. Update files also provides the user with the capability of editing existing crime information as well as deletes redundant or erroneous information. The functioning of Update files can best be understood by further decomposition of the process into sub-processes. This decomposition will be presented later in the level two Data Flow Diagrams

The fourth process which is depicted on the level one Data Flow Diagram is titled "Generate Reports". As seen in Figure 2.4, this process retrieves necessary information from the Crime Files and the Archive Reports file. It then uses this information to produce the desired report. The function of bubble 4.0 is twofold. First it must acquire the information which is necessary to produce a specific report and second, it must generate that report. In order to highlight the details of this process, it will be further decomposed into sub-processes. The results of this decomposition will be shown in the level two Data Flow Diagrams.

6. Level Two Data Flow Diagram

In Figure 2.5 we see the sub-processes which comprise bubble 3.0. They are process 3.1 labeled "GetInfo", process 3.2 labeled "Add Info", process 3.3 labeled "Edit Info" and process 3.4 labeled "Delete Info".

Process 3.1, GetInfo, has as its primary purpose the task of gathering information which will specify where a new record will be added (i.e.,in which specific crime file) or alternately which specific report will be edited or deleted. It

does this by allowing the user to choose a specific installation from the Installation file. Once this is done, the user can indicate a specific crime and date which will be used to locate a specific record within the appropriate crime file.

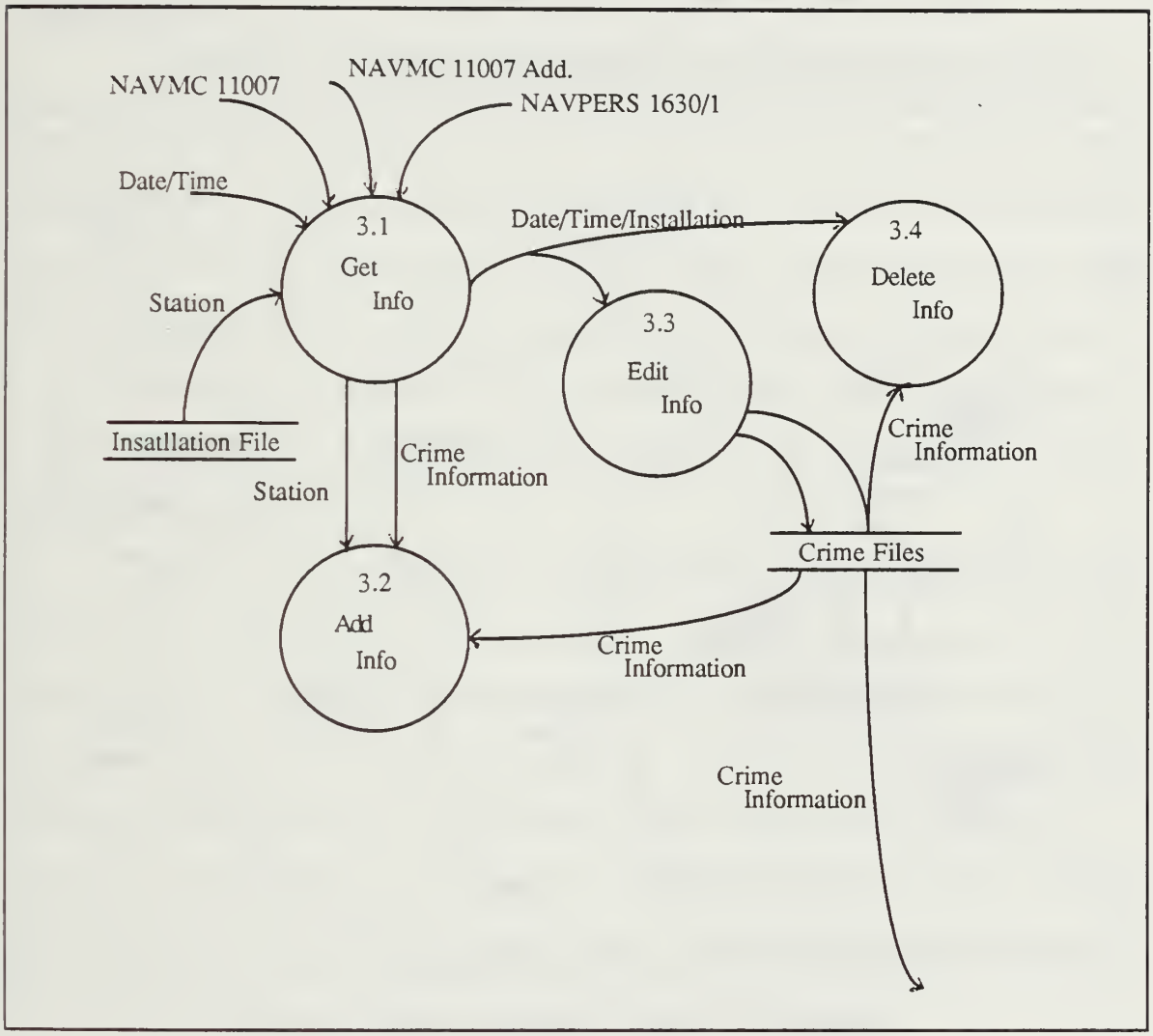

Figure 2.5 Level Two Data Flow Diagram

Add Info, process 3.2, uses the information from GetInfo as discussed above. This process will take new information (provided by the NAVMC and NAVPERS data flows) reformat it and write this information to the Crime File specified by GetInfo.

The Edit Info process, bubble 3.3, allows the user to edit erroneous data which may be stored in the Crime Files. It uses the information obtained from GetInfo.

The functioning of bubble 3.4, Delete Info, is analogous to that of process 3.3 with the exception that the specified record, once located, is deleted.

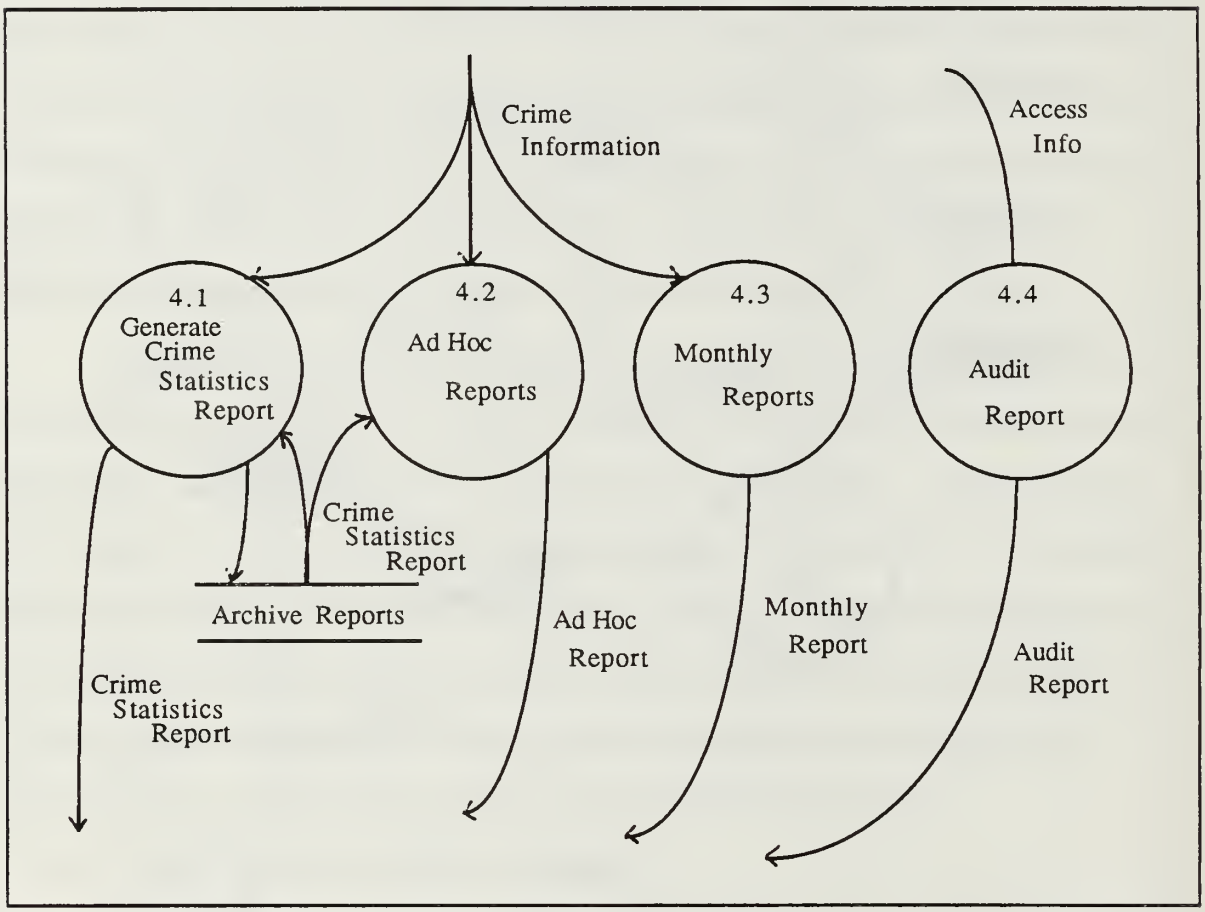

Figure 2.6 Level Two Data Flow Diagram

Figure 2.6 shows the sub-processes which comprise bubble 4.0. They are process 4.1 labeled "Generate Crime Statistics Report" process 4.2, labeled "Generate Ad hoc Report", process 4.3 labeled "Monthly Report" and process 4.4 labeled "Audit Report".

In process 4.1 the data flow "Crime Information" is transformed by "Generate Crime Statistics Report" (process 4.1) in to a data flow titled "Crime Statistics Report". This process also reads from, as well as writes to, the Archive Reports File.

Process 4.1 extracts information from the Crime File and produces yearly totals for each installation. Next it reads the Archive Reports File to obtain information about the previous years totals. With these two pieces of information, it produces a Crime Statistics Report. Lastly, it will write the current years totals to the Archive Reports File to be used as a basis for comparison with future years.

At this level of detail we can say that process 4.1 probably represents a functional primitive. The functioning of process 4.1 is described in the minispec found in Appendix B.

Process 4.2, Generate Ad hoc Report, can use either the data flow "Crime Information" or the data flow "Crime Statistics Report", or both, to generate an ad hoc report. This process searches the appropriate file (or files) for information relative to a specific installation or installations. Once the files have been accessed, the appropriate information is extracted and the required report is formatted. Process 4.2 represents the functioning of the Data Manipulation Language which will be a part of the application program used to implement the proposed system.

III. PHYSICAL DESIGN

Once we have completed our logical representation of the proposed system, our focus shifts from the "what" of the systems functioning to the "how" of the systems functioning. Because of this transition, we usually have to make a number of assumptions which simplify the development of our Data Flow Diagram. We do not attempt to show how the data is processed or how the data flows between the various processes. The Data Flow Diagram does not show error handling routines or how the system will deal with exceptional conditions. Neither does it show how normal file maintenance procedures (opening and closing files) will be handled. The purpose of the Data Flow Diagrams are, according to Davis, ..."to describe what happens without worrying about how it happens." [Ref. 2:p. 282] Our Data Flow Diagrams and mini specs are used to describe what is necessary for the system to function; Structure Charts aid us in describing how the system will function and how specifically, it should be implemented.

A. THE STRUCTURE CHART

A Structure Chart is used to show the partitioning of the system into modules. It graphically portrays the hierarchy into which the modules are arranged and the interfaces among modules. Because it is a tool of the logical design process, it does not tell anything about the decision structure of the system or the order in which the various functions are performed [Ref. 5:p. 314]. The Structure Chart shown in Figure 3.1 represents the proposed system in general terms. It is a first draft which depicts how the system will be implemented and how the source code will be structured.

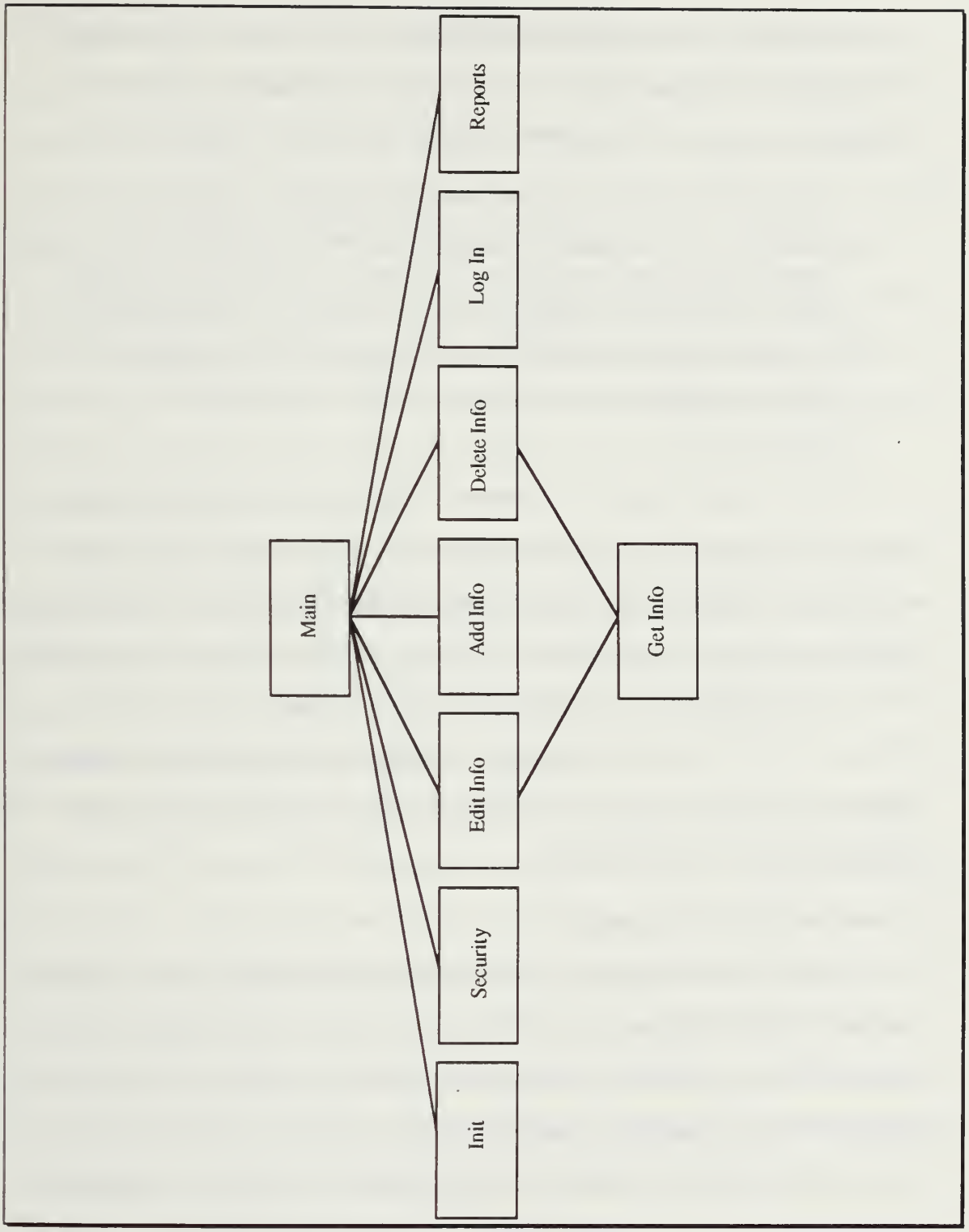

Figure 3.1 Structure Chart

Structured Analysis and design methodologies provide two methods by which we may derive a Structure Chart from the Data Flow Diagrams. They are Transaction Analysis and Transform Analysis. According to DeMarco:

> Transform Analysis applies directly to applications that are transforms --that is applications that have clearly identified input streams, central processing and output streams. Transaction analysis applies to transaction centers, parts of the application characterized by sudden parallelism of data flows. Transform and Transaction analysis are used to give the Structure Chart it's initial shape. [Ref. 5:p. 315]

As stated earlier, Figure 3.1 represents a first cut at the proposed system design. This Structure Chart will, in all likelihood, have to be altered in order to account for any deficiencies in design. These changes may be necessary because of changes in user requirements as well as data processing considerations such as the flow of control information, decision paths and error traps. Again quoting DeMarco, "There is no avoiding the fact that some revisions will be required to refine the derived structure. In fact, most of your design effort will be spent in this refinement effort." [Ref. 5:p. 319]

B. PROPOSED SOLUTION

Using the logical design as a guide, we begin to define the system in terms of processes which are implementation dependent. Aspects of the system which are nice to have but not necessary to the systems functioning and those items which are the results of implementation constraints are now addressed. In addition, modules which serve to control the flow of program execution will now be incorporated into the model.

In the Marine Corps Crime Statistics Reporting Program, the logical design dictated that program modules would have to be written which would provide security to the system, update database files and generate the required reports. These logical functions all have to be incorporated into the physical implementation of the system. The first step in meeting this requirement is to ensure that every functional primitive found on the lowest level Data Flow Diagram of the system is also represented on the Structure Chart. The Structure Chart shown in Figure 3.1 has a separate module for each of the functional primitives developed in the logical model. In this fashion, all of the logical requirements of the system will be incorporated into the source code.

Other modules will be developed which will not have their genesis in the logical design but will serve to augment functions of the modules found in the logical design. These modules are typically those that deal with program control issues. On the current Structure Chart there are two such modules, the Initialization (Init) module and the Main module.

The initialization module is an example of a program module which will be implementation specific. This module sets the parameters under which the program will operate. Considerations such as foreground/background colors, preventing inadvertent user initiated program termination, and the number of files and buffers available are all aspects of the program operating environment which should be initialized for the system. Note that the initialization module is not necessary, per se, to the functioning of the Crime Statistics Reporting Program. Because of this it did not appear in the logical design represented by the various Data Flow Diagrams, but it will be found on the Structure Chart.

The Main program is another example of a module whose etiology was in the rigor of the structured design methodologies vice logical design requirements. The main program module serves as the highest level of control within the program. It controls the flow of program execution. Its existence is only implied by the leveling of the Data Flow Diagrams but, of course, there are no control functions seen in the Data Flow Diagrams.

In summary, the Structure Chart is a tool which helps to make the transition from the logical design to the writing of the code. All of the logical functions found in the Data Flow Diagrams will be incorporated into the Structure Chart and hence into the source code. Some functions found in the Structure Chart will be the result of processes only implied by the Data Flow Diagrams. This means that there is not, of necessity, a one to one correspondence between processes in the Data Flow Diagram and modules in the Structure Chart. Indeed some processes found in the Data Flow Diagrams, Update Files for example, may seem trivial at the high level but require a good deal of code, which incorporates many distinct modules, to implement the logical requirements.

IV. DETAILED DESIGN

The process of developing a Database Management System has now progressed from a purely logical conceptualization of what the system should do to the intricacies of detailed design. We are now very much concerned with implementation issues which will impact how our logical design will be realized.

A. DATABASE ADVANTAGES

As was previously discussed in the problem definition phase of the systems development, the problem faced by the Military Police Section centered on the collection and dissemination of data obtained from individual Provost Marshall Offices located throughout the Marine Corps. The Military Police Section was charged with maintaining large amounts of data for analysis. A very high level view of this system as developed in the Data Flow Diagrams, indicated that we should develop a very large file system which would provide the user with the ability to access, manipulate and format vast amounts of data easily and efficiently. That is in fact what every Database Management System does--it automates our filing cabinet!

The many advantages of a Database Management System become evident in the context of this application. The creation of a Database would facilitate the storage and organization of vast amounts of data into a very small amount of space. As noted by Kroenke, "Data integration offers several important advantages. First and foremost database processing enables more information to be produced from a given amount of data." [Ref. 6:p. 3] Another important advantage of database processing is the elimination or reduction of data duplication. Additional benefits

of a database system according to Kroenke were; better data management, program/data independence and representation of record relationships [Ref . 6:p. 3]. It was felt that the Military Police Section could greatly benefit from the advantages of "going database" and the design of a system was begun.

B. DATA MODELING

In much the same fashion that we used our Data Flow Diagrams to model the functioning of our proposed system, we will now begin to employ database models in an effort to describe the structure and processing of our database. As noted by Kroenke:

> There are two reasons for studying database models. First, they are an important data base design tools. Database models can be used for both logical and physical database design--much as flowcharts and pseudocode are used for program design. Second, database models are used to categorize DBMS products. [Ref. 6:p. 191]

The process of moving from a logical representation of our database system to the physical implementation of that system is analogous to the progression from our Data Flow Diagrams to our Structure Chart. Kroenke describes a continuum of data base models which, on the one end are purely logical representations, and on the other end are DBMS specific models. [Ref. 6:p. 191] This continuum is reproduced as Figure 4.1 titled "Kroenke's Spectrum". Using this continuum as a guide, we will progress from a logical view of our data processing needs to a model which describes an implementable form.

1. Semantic Data Model

The first model found on Kroenke's spectrum is the Semantic Data Model. This model is chiefly concerned with ..."providing a vocabulary for expressing the

meaning as well as the structure of the database data." [Ref. 5:p. 193] As such, it is concerned primarily with a description of the data and the relationships between them. It is a good vehicle for the designer to use to communicate his design intentions to the user. The problem however, is that currently there are no Database Management Systems based on the Semantic Data Model. [Ref . 6:p. 194] Because this model could not recommend a Database Management System, the design effort moved on to the Entity-Relationship model.

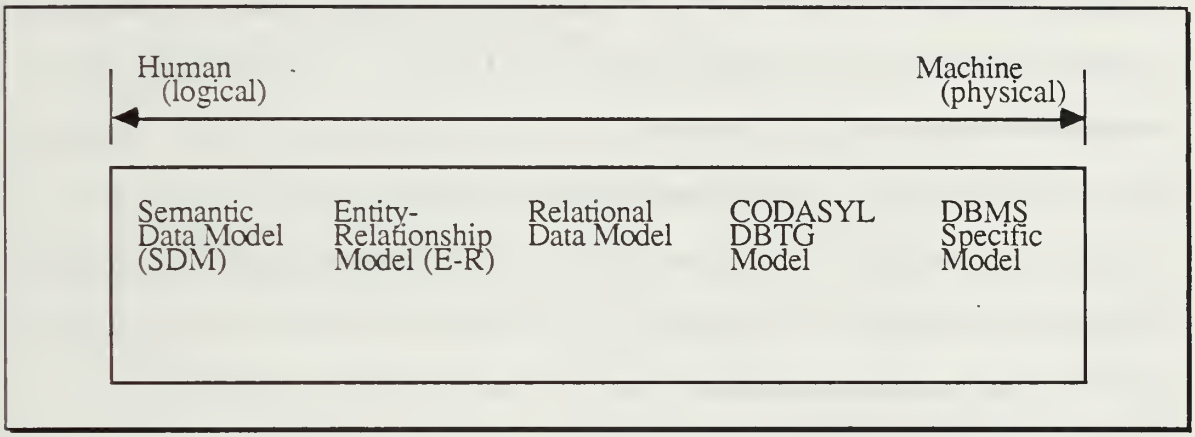

Figure 4.1 Kroenke's Spectrum

2. The Entity-Relationship Model

The Entity-Relationship model is, like the Semantic Data Model, a logical database model. However, unlike the Semantic Data Model, the Entity-Relationship model incorporates aspects of a physical model as well. The Entity-Relationship model uses diagrams to graphically portray the real world situation. Entities are the "things" in the real world which we are attempting to model. In the case of the Marine Corps Crime Statistics Reporting Program the entities which we would like to model are Offenses, Victims, Perpetrators and Installations. Each of

these entities is described by a collection of attributes. The attributes are essentially adjectives which further describe the entity.

If two entity sets are associated in some manner, they are said to have a relationship between them. In the same way in which entities can have attributes which describe them, relations can have attributes which further describe the relationship.

An Entity-Relationship Diagram is a graphic portrayal of the entities and the relationship between the entities. There are three distinct types of relationships which may be found on an Entity-Relationship Diagram. A single entity which is related to one and only one other entity from a different entity set is said to have a one to one relationship. A single entity which is associated with more than one entity of another entity set is said to have a one to many relationship. If multiple occurrences of entities are associated with multiple occurrences of entities from another entity set, the relationship is said to be many to many. The Entity-Relationship Diagram represents a convenient way not only to model all the "players" in our database system but it also allows us to model the relationship between those players.

What follows is a discussion of the Entity-Relationship Diagrams that were developed to model the enterprise known as the Marine Corps Crime Statistics Reporting Program. The methodology which underlies development of an Entity-Relationship Diagram was summarized by Davis and Olsen who stated that to develop a user data view of the enterprise and the document that view we should:

> 1. Start by identifying entity set (entity types) that are significant for his or her view of the enterprise. Each entity set is given a name and enclosed in a rectangle.

2. Relationship sets (relationship types) are identified. Each is given a name and graphed with a diamond. The entity sets that participate in the relation are indicated by connecting with an arc. The relationship is specified as 1, M, or N.

3. The attributes that establish value for the entity set are elicited and diagrammed with arcs that connect attribute to sets.

4. The process continues until all entity sets, relationship sets, attributes and value sets have been elicited and diagrammed. The diagram is a tool which allows the analyst and user to discuss the data model and resolve differences. [Ref. 1:p. 521]

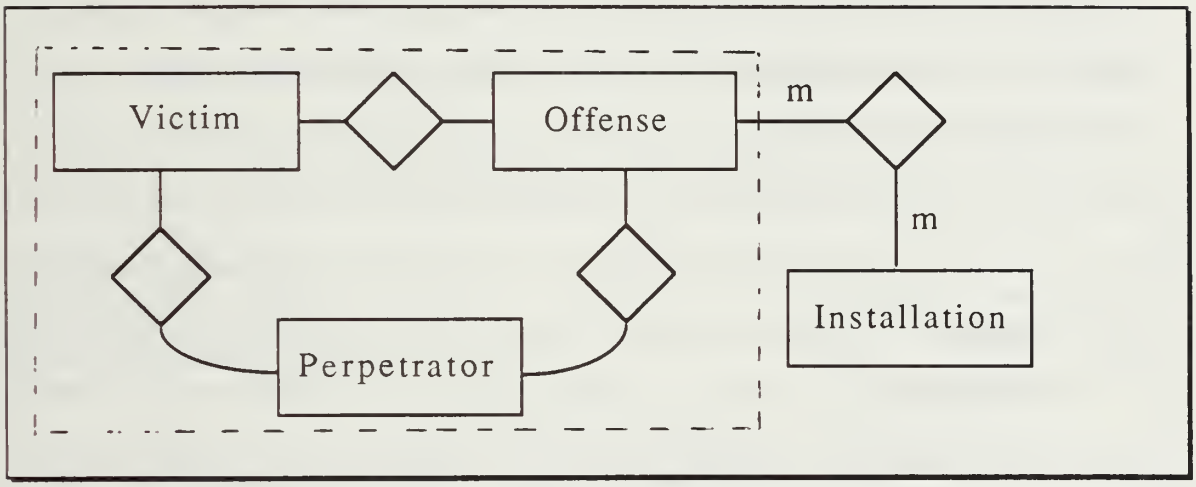

Figure 4.2 Entity Relationship Diagram

Figure 4.2 shows the entity sets and the relations between the entity sets which comprise the enterprise called the Marine Corps Crime Statistics Program. The entity types which are significant for our view of the enterprise are Victims, Perpetrators, Offenses and Installations. Between each of the entity sets are the relations which are defined by the entity sets under consideration.

Before any further attempt is made to refine our logical model; one critical observation must be made. In the enterprise called the Marine Corps Crime

Statistics Reporting Program, the entity sets "Victims" and "Perpetrators" are only modeled in the aggregate. The system, as it currently does business, does not relate a particular Perpetrator to a specific crime or even to the victim involved. It merely tracks numbers for these categories of data. This reflects the way in which the user currently does business. Indeed, they are expressly prohibited from disseminating information as it relates to specific victims of crimes. While it certainly would be beneficial to be able to link a specific perpetrator to a specific victim and to be able to record other demographics about these individuals (name, age, etc.) these functions are, at present, beyond the scope of the enterprise as it currently functions. We will, however, discuss each of these relationships as separate and distinct aspects of the model in order to facilitate any future enhancements to the model.

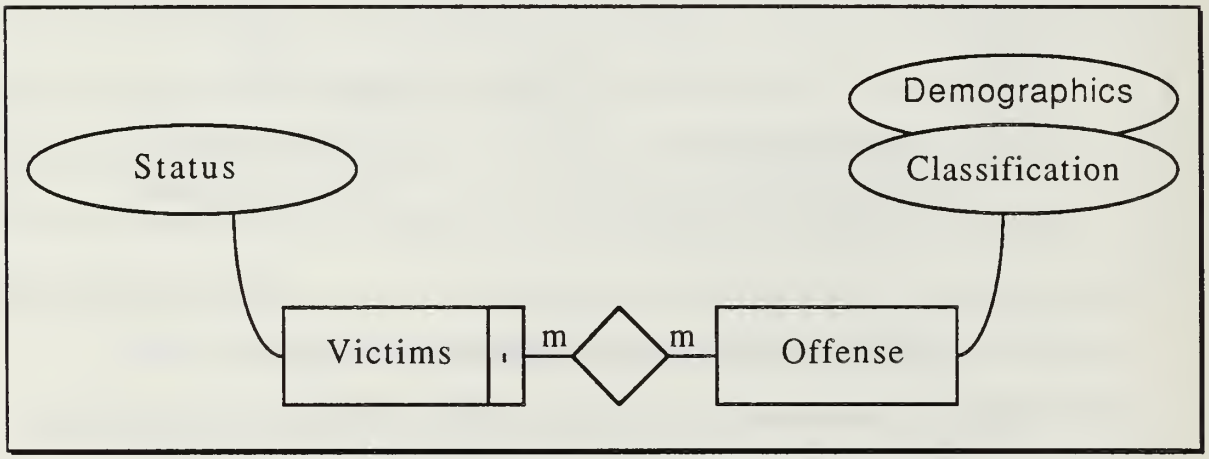

Figure 4.3 Victims-Offense Relationship

The relational tables which we will develop later in the thesis, will be based on the user view of the enterprise as modeled in Figure 4.2. The dashed line around the Offense-Victims-Perpetrator relations serves to illustrate the fact that the user is really only concerned with the relationship between Offenses and

Installations. The entities, Victims and Perpetrators, are viewed as if they were attributes of the Offense. Hopefully, future reporting requirements will allow these relationships to be fully utilized as modeled in Figures 4.3 through 4.6

In Figures 4.3 through 4.6, we have culled out each separate relation set from the Entity-Relationship Diagram proper, in order to more fully examine the relationships between the entity sets and the values which the attributes of the entity set could assume.

Figure 4.3 examines the relationship between the entity set labeled victims and the entity set labeled offense. The first thing to take note of is the fact that there exists an obligatory relationship between victims and offenses but a non-obligatory relationship between offenses and victims. Simply stated, the entity-relationship diagram is telling us that there may be crimes committed in which there are no victims (non-obligatory) but the existence of a victim presupposes and requires an offense be committed (obligatory). An example of this concept would be a Driving While Intoxicated Offense - there would indeed be an offense committed but there would not be a victim involved. The obligatory aspect of a relation is shown diagrammatically as a small dot enclosed by a rectangle within the rectangle which represents the the entity set (see Figure 4.3). Next, we should take note of the degree of the relation. In this particular relationship an offense could have many victims and a person could have been the victim of many offenses. This fact establishes the relationship between victims and offenses as being "many to many". The degree of the relation is indicated by the presence of a small letter above the line which connects the entity rectangle to the relationship diamond. A "1" indicates that a one to one relationship exists, "M" indicates a many to many relationship between the entity sets. The last piece of information which is shown on this Entity-Relationship

Diagram relates to the attributes of the entity sets and the values which the attributes may assume. The attributes ..."associate a set of values (a value set) with an entity set or relationship set and provides an interpretation of the set of values in this context." [Ref. 1:p. 519] The attributes of the Victim-Offense relationship are shown as ellipses joined by an arc to the entity set which they describe. We see that the entity set "victim" has an attribute called status associated with it and the values which the attribute "status" may assume are:

(1) USMC
(2) NAVY
(3) Dependent
(4) Civilian
(5) Male
(6) Female
(7) Caucasian
(8) Negroid
(9) Other

Likewise, the entity set offense has associated with it two distinct attributes; classification and demographics. The attribute classification can assume the following values:

(1) Murder
(2) Rape
(3) Robbery
(4) Burglary/Housebreaking
(5) Larceny/Theft
(6) Motor Vehicle Theft
(7) Drug Possession/Sales

(8) Alcohol Related Offenses

The attribute labeled demographics assumes values which are really "attributes of an attribute". By this we mean that the attribute value, murder for example, can be further described by demographics such as location, who investigated the murder, and the rate of occurrence.

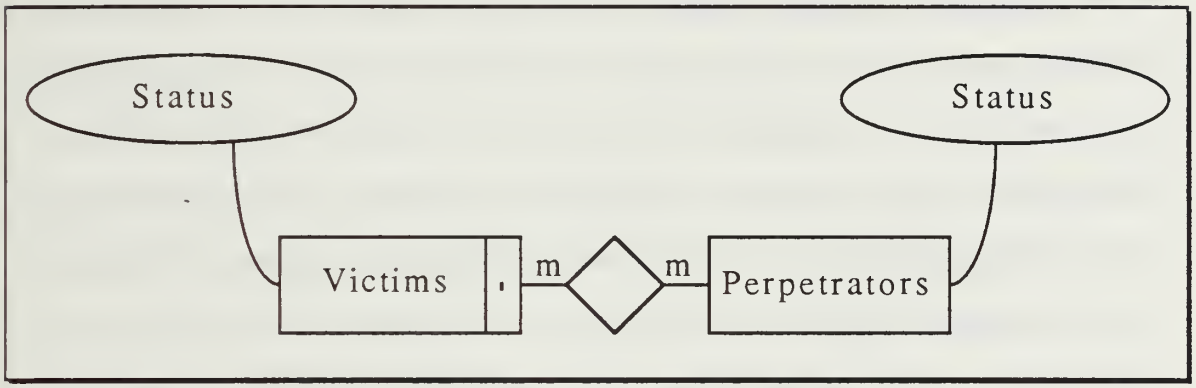

Figure 4.4 Victim-Perpetrator Relationship

In Figure 4.4, we examine the Victim-Perpetrator relationship and its associated attributes. The relationship which exists between these two entity sets shares the same characteristics as were discussed in the Victim-Offense relationship. First, there is an obligatory relationship between victims and perpetrators but a non-obligatory relationship between perpetrators and victims. This is because, as we stated earlier, the existence of a victim presupposes and requires a perpetrator of a crime for which that person is a victim. The non-obligatory relationship between perpetrator and victim follows from the fact that we can have "victimless" crimes. Next, we examine the degree of this relationship and note that it too is "many to many". Just as an individual can be the victim of many crimes, he could as easily have been the victim of many different perpetrators. It follows then, that a perpetrator could have had many victims. As we can see from figure 4.4, the

attribute which is associated with both of the entity sets is status, and the values which it may assume are the same as was discussed earlier.

In Figure 4.5, we examine the Perpetrator-Offense relationship. This relationship is unique to our enterprise in that there exists an obligatory relationship regardless of the point of view. An obligatory relationship exists between perpetrator and offense because obviously enough, to be labeled a perpetrator you must have committed some offense. The converse is also true - to have an offense committed requires a perpetrator to commit that offense. The degree of this relationship is "many to many" - a perpetrator can commit multiple offenses, and a single offense could have been committed by multiple perpetrators. The attributes which are associated with the entity sets shown in Figure 4.5 are "Status", "Classification" and "Demographics". The values which Classification and Demographics may assume were delineated in the discussion of figures 4.3 and 4.4.

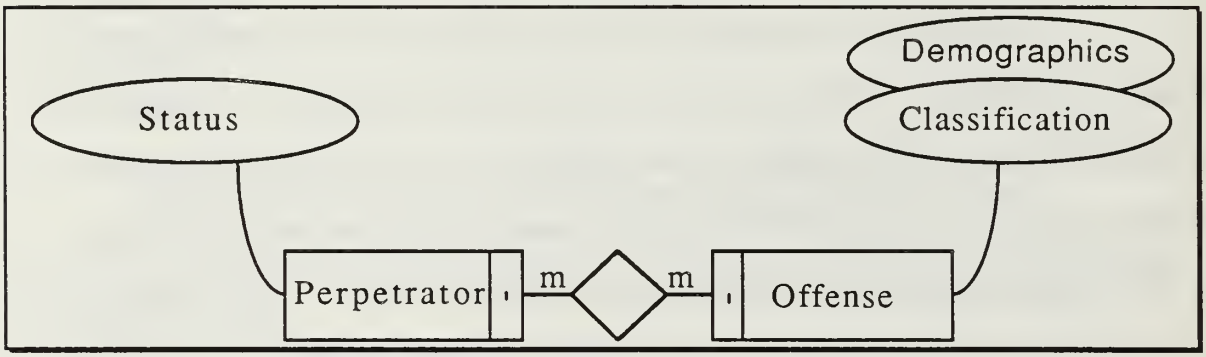

Figure 4.5 Perpetrator-Offense Relationship

In Figure 4.6, we examine the Offense-Installation Relationship. Once again we note that there exists both an obligatory and a non-obligatory relationship depending upon the point of view taken. From the Installation perspective, we have a non-obligatory relationship - the entity "Installation" can exist without inclusion in the relationship. In fact, in this instance this is a logical necessity because if it

were not so, an Installation with no crimes associated with it would not be reflected in the database. The relationship is, however, obligatory from the perspective of the Offense entity set. This is owing to the fact that the enterprise as it exists (and is modeled) is only concerned with those offenses related to some installation. It is outside the scope of the enterprise to model the fact that an offense occurred in Pokipsi , NY. We next take note of the fact that the "Offense - Installation" relationship is of degree many to many. Simply put, the model is telling us that an Installation can have multiple occurrences of an offense and conversely, a specific type of offense could occur at more than one installation. The attributes which are associated with the entity sets found in the Offense-Installation diagram are "Classification" and "Installation". The attribute "Classification" and the values which the attribute may assume have been delineated in the discussion of Figure 4.3 The attribute "Name" represents that set of possible values which will describe the Installation entity set. The following list shows the names of the twenty distinct Installations which are modeled in our Entity-Relationship diagram.

(1) MCLB ALBANY
(2) MCLB BARSTOW
(3) MCAS BEAUFORT
(4) CAMP BUTLER
(5) MCAS CHERRY POINT
(6) MCAS EL TORO
(7) CAMP ELMORE
(8) HENDERSON HALL
(9) MCAS IWAKUNI
(10) MCAS KANEOHE
(11) MCB CAMP LEJUENE
(12) MCRD PARRIS ISLAND

(13) MCB CAMP PENDLETON
(14) MCDEC QUANTICO
(15) MCRD SAN DEIGO
(16) MCAS TUSTIN
(17) MCAGCC TWENTYNINE PALMS
(18) MCAS YUMA
(19) CAMP SMITH
(20) MCAS NEW RIVER

Now that we have modeled each of the separate relations in our enterprise, we can begin to discuss the way these relations fit together. Referring back to Figure 4.2, we get a holistic view of our enterprise as modeled in one Entity-Relationship diagram. In this diagram, we see not only the separate entity relationships as discussed in Figures 4.3 through 4.6 but we are now able to see how these relations relate to each other. The most important thing to note is the fact that the entity set "Installation" is only transitively related to the entity sets Victims and Perpetrators. From the perspective of our logical design, there is nothing really noteworthy about this fact other than it exists. From an implementation aspect however, it could determine the way data stored in our database will be accessed. What it implies is that information about victims or perpetrators will probably be linked to a compound key consisting of values of attributes from the offense and installation entity sets. Whether this is true or not will be determined after the data model is further refined.

The Entity-Relationship model was used as a tool to help define and understand the data which will be manipulated by the Marine Corps Crime Statistics Reporting Program. Since there are currently no software packages which would allow us to implement a Database Management System using the constructs of the Entity-Relationship model, we will move along Kroenke's continuum of database

models (see Figure 4.1) to the relational Model. This transition is quite natural because as Kroenke noted, "Entity-Relationship diagrams can be used to express a physical design for relational implementations" [Ref. 1:p. 96].

3. The Relational Model

In 1970, E. F. Codd of IBM presented a paper that has since been acknowledged to be a milestone in data systems research. In this paper he proposed the Relational Model for data storage. This model represented a totally new concept in data storage and database design. There were two driving forces which led to the development of this model. The first was the desire to build a system that would make data storage ..."independent of any specific user viewpoint, and the second was to find a way to define data bases so that they could be treated using formal mathematical methods." [Ref. 7: p. 46]

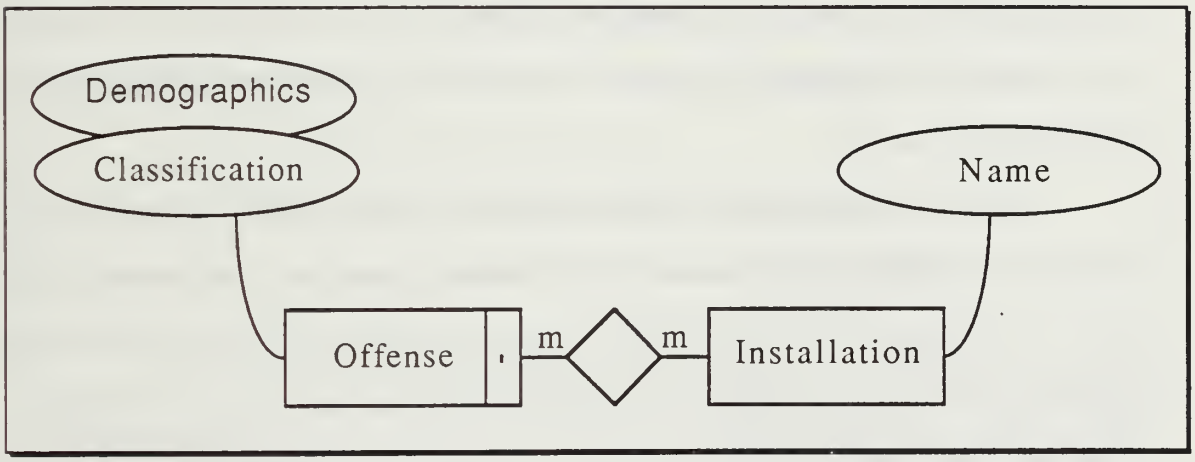

Figure 4.6 Offense-Installation Relationship

The relational model represents data in a two-dimensional table called a relation. Within each relation there are rows and columns. The rows of the relation are called tuples. The sequence of the tuples is not important, but once it has been established, all tuples must appear in that sequence. A record within the data base is

data base is stored as a tuple. The columns of the relation are termed attributes of the relation. Each attribute has a domain of values which it may assume. [Ref. 7:p 46]

To illustrate how the model was used, the Entity-Relationship diagrams developed earlier were transformed into the tables of the relational model. The relational tables which we develop will be the basis for a physical implementation of the data structures. Before we attempt this transformation we must take note of certain "normalization" rules. These rules serve to ensure that errors which could occur as a result of updating, inserting or deleting records stored in this format do not occur.

Before we can properly interpret and use these rules, some normalization terminology needs to be presented. A value Y is said to be functionally dependent on attribute X if the value of X determines Y [Ref. 6:p. 289]. An example would be the purchase price of a new automobile. The price of the car (Y) is dependent upon the make of the car (X).

The next item we need to discuss is that of a determinant. A determinant is the value of an attribute which causes the functional dependency. In the case of our automobile example, the make of the car was the determinant because it determined the price paid for the automobile. It should be noted that one (or more) attributes can be the determinant of several attributes. In this case, we would see a transitive determinacy. Again using our car example, we note that the sales tax paid is transitivity dependent on the make of the car. In other words the make of the car determined the price, and the price of the automobile determined the sales tax. Therefore, the make of the car transitively determined the sales tax.

One more point must be made. Recall that a key is an attribute which uniquely identifies a tuple. As such, the value of the key must be unique, but the value of a determinant may or may not be unique.

Now that we have a grasp of the terminology used in the normalization process, we will define the normal forms .

a. First Normal Form

Under first normal form all occurrences of a record must contain the same number of fields.

b. Second Normal Form

Under second normal form a non-key field must provide a fact about the key, the whole key and nothing but the key.

c. Third Normal Form

A relation is in third normal form if it is in second normal form and every non-key attribute is non-transitively dependent on the primary key.

d. Boyce/Codd Normal Form

A relation is in Boyce/Codd Normal Form (BCNF) if and only if it is in third normal form and every determinant is a candidate key.[Ref . 8:p. 121].

The other normalization forms; Fourth, Fifth and Domain Key Normal Form are more restrictive forms which govern join and functional dependencies.

We will attempt to ensure that the relation table which we derive from our Entity-Relationship diagram is in Boyce Codd Normal Form. This will help to ensure that we eliminate as many duplications as possible as well as to ensure that we will suffer no insertion or deletion effects when the database is updated.

Offense	Date	F-1	F-2	F-3	F-4	F-25	F-26	F-27	F-28
Murder	11/86	2	4	5		6	0	7	3
Rape	2/87	6	3	2		4	9	4	5

Figure 4.7 Relational Table

Figure 4.7 represents the relational table which we derived from the Entity-Relationship diagram of Figure 4.2. Note that in the First Normal Form all we have attempted to do is relate the attributes of the entities in a tabular format. We can say that this table is in First Normal Form because all of the tuples have the same number of attributes. In other words, there are no "holes" in the table; each record has the same number of fields. Since relations in First Normal Form suffer modification anomalies, we will continue to refine our relational table. We next consider Second Normal Form to help eliminate some of these anomalies. In our relational table each tuple is uniquely identified by the compound key Crime-Date. Every non-key attribute, designated as F-1 through F-28, (see Figure 4.8 for a descriptive summary) provides a fact about the compound key. Therefore, our table is also already in second normal form. The other normal forms all address anomalies caused by functional dependencies [Ref. 6:p. 289].

In our relational table all the non-key attributes are functionally dependent on the primary key. Since it is true that the table is already in Second Normal Form, the relationship is also in Third Normal Form. In order to be in Boyce/Codd normal form our table must satisfy the constraints of third normal form and every determinant must also be a candidate key. Since the only determinate is our compound key, our relation is also Boyce/Codd normal form.

It should be noted that the ease with which this table normalizes to Boyce/Codd Normal Form is the exception to the rule. Had we needed to normalize all the relations as delineated in Figures 4.3 through 4.6, there would have been many more tables to consider and a great deal more projections needed to normalize to Boyce/Codd Normal Form.

FIELD	FIELD NAME	DESCRIPTION
1	F-1	offenses on base
2	F-2	offenses of base
3	F-3	unfounded false reports on base
4	F-4	unfounded false reports off base
5	F-5	cleared by arrest
6	F-6	investigated by NIS
7	F-7	rate per 1000 personnel assigned
8	F-8	total this month last year
9	F-9	perpetrator - USMC
10	F-10	perpetrator - other services
11	F-11	perpetrator - dependent
12	F-12	perpetrator - DOD personnel
13	F-13	perpetrator - male
14	F-14	perpetrator - female
15	F-15	perpetrator - Caucasian
16	F-16	perpetrator - Negroid
17	F-16	perpetrator - all others
18	F-17	victim - USMC
19	F-18	victim - other services
20	F-20	victim - dependent
21	F-21	victim - DOD personnel
22	F-22	victim - male
23	F-23	victim - female
24	F-24	victim - Caucasian
25	F-25	victim - Negroid
26	F-26	victim - all others
27	F-27	drug involvement
28	F-28	alcohol involvement
29	CRIME	classification of offense
30	DATE	date of offense
31	CODE	reserved for future use

Figure 4.8 Descriptive Summary of Data Fields

C. SOFTWARE REQUIREMENTS

The process of developing a Database Management System has now progressed from a purely logical conceptualization of what the system should do to the

intricacies of detailed design. We are now very much concerned with implementation issues and how our logical design will be realized. From our feasibility study we learned that this Database Management System could be implemented in a microcomputer environment. Considerations such as the operating system, the type of microcomputer, the storage medium and the software package which we would use to develop our system were all implementation issues which could not be addressed until a logical definition of the systems functionality was established. Armed with this understanding of what the system should do, we are now ready to formalize how the system should do it.

The software industry has seen an explosive growth in the past few years and as a result , the number and quality of Database Management Systems commercially available has dramatically increased. In fact, one trade publication stated that "1988 will be a watershed year for Database Management software" and went as far as to proclaim this "The year of the Database."[1] With such a plethora of software available to the developer, choosing the right Database package would seem to be an almost intractable proposition. These products span the spectrum of power and price. Deciding which Database Management System to use to implement the Marine Corps Crime Statistics Reporting Program would have to be based on the best combination of many different factors. From the perspective of the systems developer, I was primarily concerned with obtaining a package which enjoyed widespread use, supported the relational model and would allow for portability of the data of the Marine Corps Crime Statistics Reporting Program. There are a number of features that could be used as a basis for ranking competing software

---

[1] Personnal Computing, January 1988

packages. The following were suggested by a study conducted by the Information Technology Services of Stanford University:

(1) functionality - how does the package support data entry, data exchange, report writing and programming?

(2) data types - does the product support floating point, fixed point decimal or integer, date, time string, monetary, Boolean and other types?

(3) reporting - does the product have an easy to use report definition language with the ability to generate form letters, columnar reports and reports of an arbitrary format?

(4) data entry capability - does the package require a programming language to be able to do form data entry?

(5) query capabilities - how well does the software do data management tasks (i. e. , does it support the relational model)?

(6) documentation - is the product supported by manuals which are well written and have many examples?

(7) reference documentation - is there a complete and concise summary of topics and functions?

(8) interface - can the product exchange data in DIF, BASIC, Fixed ASCII and other formats?

(9) ease of use - how difficult is it to install, configure and learn to use the product?

(10) familiarity - do we have any previous experience with this or a similar product? [Ref. 9:pp. 41-49]

The considerations listed above are focused towards the developer of the system. We must also address, however, whether or not this software package will allow us to develop a system tailored to the needs of the end user. As noted in the Stanford study, selecting the best Database Management system in this context means ..."finding the best match between a potential group of software packages and

two other factors, the users skill (patience, attitudes and perceptions) and the users needs (the requirements the user is going to expect the software to meet)". [Ref. 9:p. 9]

The typical user of the Marine Corps Crime Statistics Reporting Program will be what is characterized as end user or support staff. They would be the individuals who typically were responsible for the filing and storage of records in the manual system. They would be responsible for the preparation of printed reports and clerical activities. As such, we would need a software package which would allow the development of a system which is "user friendly".

Systems designed for this type of user should have characteristics which ensure ease-of-use. The Information Technology Services of Stanford University suggested the following as a guideline for developing a user friendly system:

(l) They should be menu-driven, since use may be infrequent and training time must be low (since turnover or rotation of responsibilities may be high.)

(2) They should have on line help and/or documentation, since the user cannot be expected to spend an hour or more reading manuals before using the system.

(3) They should offer good error messages and informational messages, leading to quick diagnosis and correction of any error.

(4) They should have simple technical operation and visual presentation involving clear and consistent keyboard and function key use (with keyboard or on-screen labels), little or no disk manipulation, and clear layout of screens. [Ref. 9:pp. 13-14]

After surveying many of the Database Management Systems currently available, I chose to implement the system in DBase III Plus. This decision was made based on all of the criteria discussed, as well as issues such as vender support, price and portability.

The overall concept is to provide a system to the user that is extremely easy to use yet still retains a great deal of power and flexibility. DBase III Plus will give us the ability to build a menu driven system that will produce custom formatted reports yet remain flexible enough to evolve with changing user requirements.

D. HARDWARE REQUIREMENTS

Now that we have determined which application tool will be used to implement the Marine Corps Crime Statistics Reporting Program, we are ready to discuss the hardware requirements. At this juncture we are only concerned with those pieces of hardware which are essential. Items which will enhance system performance will be addressed later as recommendations.

The following pieces of hardware are necessary at a minimum:

(1) An IBM PC, IBM XT, IBM AT or 100% IBM compatible with at least 256 kilobytes of random access memory.

(2) A disk operating system, PC-DOS or MS-DOS version 3.0 or later.

(3) Two 360 kilobyte disk drives.

(4) A printer of at least 80 column capability.

V. DOCUMENTATION

The source code which will be discussed presently represents the culmination of our systems analysis and design exercise. Starting with problem definition, we moved through a feasibility study, requirements analysis, logical design phase, detailed design phase and finally to an implementation phase. During each phase the picture of the what our system would eventually look like was gradually brought into focus. In every phase there was some deliverable which served as a guide to our implementation. The requirements analysis and logical design phases indicated what needed to be done. They provided us with a high level view of the system as represented by the Data Flow Diagrams and our first cut Structure Chart. The detailed design phase culminated with the relational table. This table served as the basis for the physical structure of our files. Finally, the implementation phase ended with our system as depicted in Figure 5.1.

All of the deliverables of our systems design and analysis effort laid the foundation for a system which is thoroughly documented. While they represent a good start, they do not necessarily guarantee a maintainable system. As noted by Pressmen, "Maintainability may be defined qualitatively as the ease with which software can be understood, corrected, adapted and/or enhanced." [Ref. 10:p. 34] In order to facilitate the understanding of the software developed for the Marine Corps Crime Statistics Reporting Program, we augmented the source code with additional supporting documentation.

Appendix A contains the Data Dictionary. This provides the user with a convenienent place to look up definitions of terms they may not understand. It will

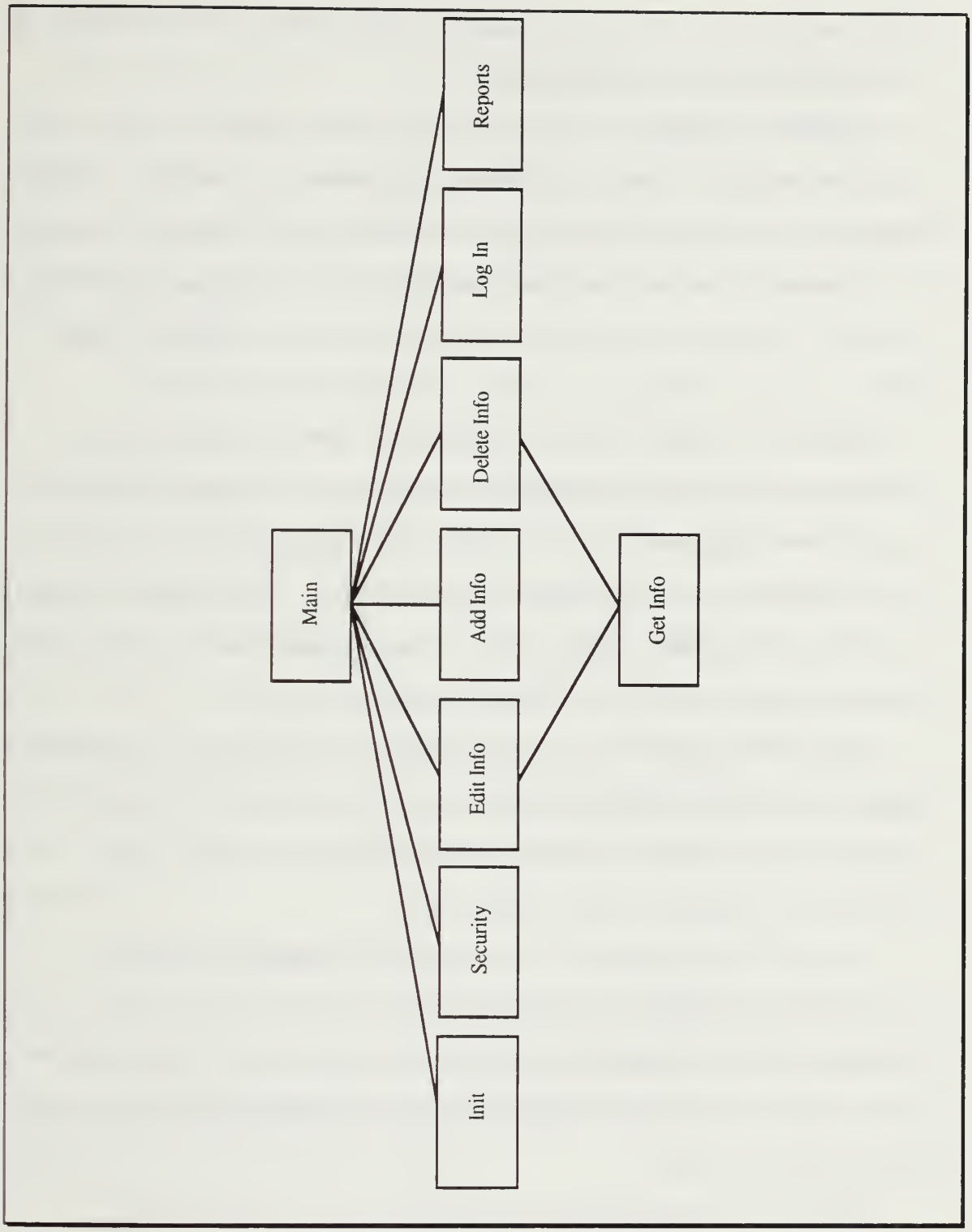

Figure 5.1 Structure Chart

contain definitions for all of the components found on the Data Flow Diagrams as well as those found in our database files.

Appendix B contains the mini specs for the primitive functions found on our Data Flow Diagrams. These are plain English statements that delineate what each functional primitive does, as well as specify the relationship between the data flows.

Appendix C contains several diagrams which show how the database files are structured. Information such as the lengths of the different fields within each record, the type of fields and key fields is shown.

Appendix D contains "Control Structures" for each of the major modules within the program. These are essentially diagrams of the modules which show how the levels of logic are imbedded within a given module. Only the major control statements, "Do while" loops, "Case statements", "If statements", etc.,are shown on these diagrams. Utilizing these diagrams the user can see, condensed on a single page, the structure of code that may span several pages.

Appendix E contains the source code listing. The beginning of each module listing will show all memory variables used within each module, the calling module, the called module, input files and output files. In addition, a brief explanation of the purpose of the module is given.

In addition to the appendices, the last part of this chapter is dedicated to explaining the functionality of each of the modules in the program. It is hoped that this documentation will enable future users of the system to be better able to correct, adapt or enhance the software as the software's environment changes and user requirements expand.

A. DATABASE STRUCTURE

The Marine Corps Crime Statistics Reporting Program utilizes 22 different database files to function. As was mentioned earlier, twenty of these files are use to store the monthly reports for each of the individual Marine Corps installations. The Audit database file stores the yearly totals, by installation, which are compiled by the program. The Archives database file stores information about the users who accessed the system.

1. Installation Database Files

All of the installation files share the exact same format. Each consists of 31 different fixed length fields with one record assuming 94 bytes of storage space. The primary key for any record within the database file is a composite of the "crime" and "date" fields. The fields labeled F1-F28 correspond to the same "fields" on the "Monthly Police Activities Report" (NAVPERS 1630/1). The alpha-numeric format is used do designate these fields just as a matter of convenience. For the descriptive titles which correspond to the alph-numeric designator, see Figure 4.9. The structure of the Installation database files, the width of specific fields and the field types are diagrammatically illustrated in Appendix C.

2. Archives Database File

The Archives Database File is the repository for the annual figures generated by the "Compile Annual Report" option found within the program. It's structure was defined by the need to report specific crime statistics as well as statistics about entire categories of crime.

Referring back to the Monthly Police Activities Report (NAVPERS 1630/1), Figure 5.2, we see that the first column is labeled "Classification of the

PART I INDEX CRIMES

CLASSIFICATION OF OFFENSE	OFFENSES KNOWN OR CASE	OFFENSES KNOWN OR CASE	FALSE REPORTS OFF ON CASH	FALSE REPORTS OFF OFF CASH	Cleared by Arrest	Cleared by Arrest	Investigated by NIS MCI	Rate per 100,000 Mil Pop Assigned	This Month Last Year	STATUS OF PERPETRATOR USMC	STATUS OF PERPETRATOR NAVY	STATUS OF PERPETRATOR DEP	STATUS OF PERPETRATOR CIV	STATUS OF PERPETRATOR MALE	STATUS OF VICTIM Female	STATUS OF VICTIM CAUC	STATUS OF VICTIM NEG	STATUS OF VICTIM ALL OTHER	Involvement Drug Alcohol
	1	2	3	4	5		6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1 HOMICIDE																													
A MURDER																													
B MANSLAUGHTER																													
2 RAPE																													
A RAPE																													
B ATTEMPTED RAPE																													
3 ROBBERY																													
A FIREARM																													
B OTHER WEAPON																													
4 ASSAULT																													
A FIREARM																													
B OTHER WEAPON																													
C SIMPLE ASSAULT																													
5 BURGLARY																													
A FORCIBLE ENTRY																													
B ATTEMPTED ENTRY																													
C GVT PROPERTY																													
D NON-GVT PROPERTY																													
6 LARCENY THEFT																													
A GVT PROPERTY																													
B NON-GOVT PROPERTY																													
7 MOTOR VEH THEFT																													
A AUTO																													
B OTHER																													
C GVT PROPERTY																													
D NON-GVT PROPERTY																													
8 NARCOTICS MGT																													
A USE & POSSESSION																													
B SALE & TRAFFICKING																													
9 MARIJUANA																													
A USE & POSSESSION																													
B SALE & TRAFFICKING																													

Figure 5.2 NAVPERS 1630.1

Offense". Note that the crime of "Homicide" is actually composed of the totals for murder and manslaughter. The program accepts murder and manslaughter as separate entries but will use memory variables to sum them together. This total will be stored in a memory variable labeled "Homicide". The program will do the same sort of thing for the other reporting categories. For example, the category "Index Crimes" consists of the subcategories Violent Crimes and Crimes Against Property". However, these two subcategories are further divided into the specific Offenses of Homicide, Rape, Robbery, Aggravated Assault, Burglary, Larceny and Motor Vehicle Theft.

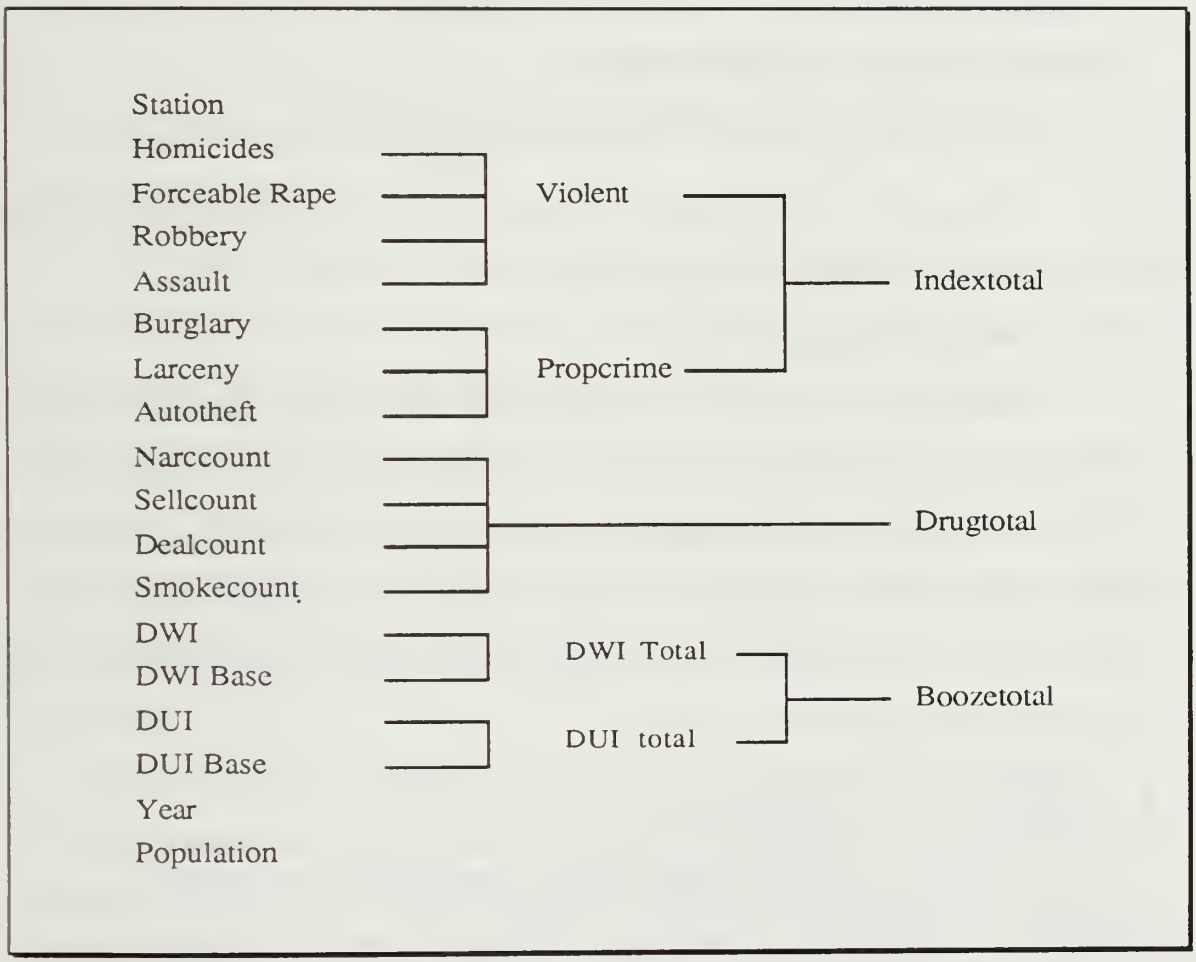

Figure 5.3 Archives Database Structure

The program will perform the calculations needed to arrive at the totals for the various categories of crime and store it to the appropriate field in the Archives Database file.

Figure 5.3 shows the twenty three different crimes tracked by the system. This tree like diagram shows which fields are combined to yield the totals for the different categories of crime. The major categories reported and the fields which store these values within the Archives database file (as indicated in Figure 5.3) are :

(1) Index Crimes - stored as Indextotal
(2) Violent Crimes - stored as Violent
(3) Crimes Against Property - stored as Propcrime
(4) Drug Offenses - stored as Drugtotal
(5) Alcohol Related Driving Offenses - stored as Boozetotal

The structure of the Archives database file, the width of specific fields and the field types are delineated in Appendix C.

3. Access Database File

Each time a user accesses the system, his name, rank, unit, date and time of access, and any menu options chosen will be stored to the Access database file. This file will provide the user organization with an audit trail of transactions on the system. Each record will consist of six fields which will occupy 53 bytes of total storage space. The structure of this database file, the width of specific fields and the field types are illustrated in Appendix C.

B. SOURCE CODE SUMMARY

The Marine Corps Crime Statistics Reporting Program was designed specifically to produce the Annual Crime Statistics Report. This report consists of

information from twenty Marine Corps Installations pertaining to 26 different types of crime. The information is presented in four different report formats with each format having a prior year - current year comparison made. In addition, totals for each year, percent change between years and crime rate per 1000 individuals is calculated. Although not specifically required a monthly report format was also included.

Before delving into the functioning of each specific module, an overview of how the program functions will be presented. The program has 22 separate database files; one for each of the twenty reporting installations, one for the Archival database and one for the Access data base. When a user accesses the system, he chooses an installation to work with. This choice determines which of our data base files will be accessed. From here he can edit information relevant to a specific installation.

In the Reports module the user can specify whether a Monthly Report, an Audit Report or an Annual Report is required. If an Audit Report is required the system accesses the Audit database file and provides information as to who accessed the system and when. If the user wishes to print an Annual Report he simply selects that option. The program will access the Archives database file and get all of the records pertaining to all of the reporting installations for the current year as well as the previous year. This information will be formatted appropriately and output to the printer.

If the user selects "Compile Annual Report", the program will search each Installation database file (all twenty) for crime statistics for the year requested. It will total all the crimes into the appropriate categories and append this information as a new record to the Archival database file. How these functions are implemented

specifically and the error traps incorporated into each module will now be discussed.

1. Security Module

The purpose of the Security module is to allow only authorized users to access the Main program. It does this by having the user enter his name, rank, unit and password. If after three tries, he fails to enter the correct password, the program returns control to the operating system.

This function is accomplished in the code as follows. (See Module Description Summary Page, Appendix D) The first "IF" statement embedded within the "Do while" loop checks for a match on the password entered against the password which is coded into the program. If the password entered matches that in the code, the Main program is called. If the password entered does not match, a counter variable (loop) is incremented by one. Next, the value of the counter is checked by the second "IF" statement found within the "Do while" loop. If the counter is less than 3, the program exits the "If" statement and returns to the "Do while right" level. This of course returns us to the first "If" statement and the user can now enter his second try at the correct password. If this attempt also fails, the loop counter is again incremented. Since its values still checks less than three, the "Do while right" level passes control back to the first "IF" statement and the user can enter his third try at the correct password. If he again fails to enter the correct password, the loop counter is incremented to a value of three. Now the condition tested for by the second "IF" statement tests true and the "If loop = 3" statement is executed. This causes a message to be presented to the user which tells him that he may not access the system. Embedded within the "If" statement is the "Do while delay < 50" loop. The purpose of this loop is to suspend program execution so that

the message which is presented to the user remains on the screen long enough to be read. This simple mechanism is used extensively throughout the program to temporarily display error messages as well as other prompts to the user. After the program circles through the delay sequence fifty times, the delay loop is exited and DBASE III Plus executes the "quit" command. This returns the user back to the operating system of the microcomputer.

2. Main Module

The function of the Main program is to allow the user to chose an option and to set the initial operating environment of the program. This simple program consist of a case statement embedded within a "Do while" loop. The "Do while" loop is simply an error checking scheme, in that it will cause the program to continue to present the main options menu until the user selects a value (1-5) which is listed. The case statement will execute the appropriate program based on user selection. If a value is entered which is out of a range, the case statement causes an error message to appear in a window overlaid on the menu. It then passes control to the "Do while" loop and the main menu is presented once again. The manner in which the Main program sets the initial operating environment is by making a call to the Init.Prg module.

3. Init Module

This program simply sets a number of DBASE III Plus defaults such as foreground and background colors on the screen. DBASE III Plus memory status reporting is suppressed as well as all of the PF Key settings and the escape key. Most of this was done to prevent access to the program from the DBASE system level and to help prevent inadvertent user initiated program termination. It also ensures that all database files are closed and all memory variables are released

prior to main program execution. Once the parameters are set, control is returned to the Main (calling) module.

4. GetInfo Module

This module is used by both Edit.Prg and Delete.Prg. Its function is to allow the user to specify a record which will be edited or deleted. When this program is called, the first "Do while" loop (Do while exit ="N") tests true and control is passed to the lower level "Do while choice > 20" loop. This loop will present the user with a list of all twenty Marine Corps Reporting Installations. The user then selects the number corresponding to the Installation which contains the information that he is interested in. If the user enters an improper value (i.e. a letter or a number less than 1 or greater than 20), the "otherwise" portion of the case statement is executed, choice is set equal to 99 and an error message is displayed. The "Do while Choice > 20" loop tests true and the user is again presented with a menu of the Marine Corps Reporting Installations. This time he (hopefully!) picks an appropriate value and the case statement stores the choice to the memory variables "DBF" and "Station". DBF will be used in the calling module to access the appropriate database file. Station is used to ensure the appropriate title is used on all screens used in the calling module.

After the Station has been selected by the user, the program exits the "Do while Choice > 20" loop and executes the "Do while (month < 1) or (month > 12)" loop. This loop obtains the date of the desired report. The input is checked to ensure a valid date (in terms of number of months) is entered. The program will loop in this section of code until a valid entry is made. Once this is accomplished the date is stored in a memory variable called "Date". This memory variable will be part of the Key, used by calling program, to search for the appropriate record.

Once an appropriate date has been entered, the program exits the "Do while Month" loop and executes the "Do while choice > 3" loop. This loop presents a menu of Crime Classifications which the user may select to work with. If the user makes an inappropriate entry, the "otherwise" portion of the "Do case" statement (which is embedded in the "Do while choice > 3" loop) will cause an error message to be displayed. After the error message is displayed control returns to the "Do while Choice > 3" loop and the Menu of Crimes Classification is presented to the user once again.

If the user makes an appropriate selection, the "Do Case" Statement will invoke the appropriate program. These programs, Indxcrms.Prg, Drugs.Prg and Alcohol.Prg are just lower levels of menus which delineate specific crime choices to the user. The choice made in one of these programs is stored as a memory variable called "Crime" and this value is returned to the calling program (GetInfo) when control is return to GetInfo.

Once we have all of the necessary information (Station, Crime and Date) the program executes the "If verify = N" statement. This shows the user the values of the information which will be used to search the database for the record he has specified. If all of the information is deemed correct, the program stores "Y" to the memory variable "exit". This causes the "Do while exit = N" loop to test false and control is passed back to the calling module. If the user notes an error and enters anything other than "Y", "N" will stored to the memory variable "exit" and "Do while Exit = N" will test true. This will cause the program to loop to the beginning of this module and begin the whole selection process again.

5. Edit Module

The Edit module allows the user to access any one of the twenty database files and edit information from a specific record within that database. It functions as follows. When the user chooses the Edit option from the main menu, Main.Prg passes control to the Edit module. The first statement in this module "Do while exit = N" tests true so the next statement "Do GetInfo" is executed. GetInfo returns values for the Memory variables "DBF", Crime" and "Date" when it returns control back to the Edit module. The next statement executed, "USE &DBF", is a macro substitution. It tells DBASE to open the database file indicated by the value of the DBF memory variable. Next, the program uses a locate command to find the specific record within the database file. This statement uses a macro substitution in order to search the data base file for a specific record based on the date and crime which was obtained in the GetInfo module. If the program fails to find a record based on the crime and date fields, (i. e. "If not found() and EOF() tests true) a message is displayed requesting that the user verify his input. Control now goes back to the "Do while exit = N" loop, GetInfo is called again and the process repeats itself.

If a record is located, the program executes the "Else" portion of the "If not found () and EOF ()" statement. The "Do while Repeat = Y" loop embedded within the "Else" clause, tests true. This causes the program to store all the field values of our selected record into memory variables. Next, a screen is generated which shows the user all the information contained in that record. At this point "get" statements allow the user to change the value of any of the memory variables which are displayed on the screen. After all the changes have been made, the user is asked to verify his changes. If the changes he made are correct, "If verify = Y"

tests true and the new values are written into the record by the "replace" commands. The program then exits all of the loops, closes the database, releases all the memory variables and returns control back to the calling (Main) program.

If an error has been made, the user enters anything other than Y when prompted to validate his input. This causes the "If Verify = Y" statement to test false so that the Else clause of the statement is executed. This stores a value of "Y" to the loop control variable "repeat". The "Do while repeat = Y" statement now tests true and the user is once again presented with a screen showing the field values of the record he is working with. The user will continue to loop in this part of the module until he verifies that all entries are correct. The program will then exit to the calling module as discussed above.

6. Delete Module

The Delete module allows the user to access any one of the twenty installation database files and locate a specific record (or records) for deletion. It functions as follows. When the user chooses the Delete option from the main menu, Main.Prg passes control to the Delete module. The first statement in this module "Do while exit = N" tests true so the next statement "Do GetInfo" is executed. GetInfo returns values for the Memory variables "DBF", Crime" and "Date" when it returns control back to the Delete module. The next statement executed, "USE &DBF", is a macro substitution. It tells DBASE to open the database file indicated by the value of the DBF memory variable. Next the program uses a locate command to find the specific record within the database file. This statement uses a macro substitution in order to search the data base file for a specific record based on the date and crime which was obtained in the GetInfo module. If the program fails to find a record based on the crime and date fields, (i. e.,"If not found() and

EOF()" tests true) a message is displayed requesting that the user verify his input. Control now goes back to the "Do while exit =N" loop, GetInfo is called again and the process repeats itself.

If a record is located, the program executes the "Else" portion of the "If not found () and EDF()" statement. This causes a deletion Menu to appear on the screen. The screen shows the key record data as well as deletion options. If the user enters a "1", all records with the same date (i.e.,month and year) as the one chosen by the user will be deleted. If the user chooses option number "2", the statement "Case Choice = 2" within the "Do case" construct is executed. Now only a specific record (uniquely identified by crime and date) will be deleted. After the deletion takes place the database is packed and closed, all memory variables are released and "Y" is stored to the loop control variable "exit". The "Do while exit = N" loop will now test false and control will return back to the Main program.

If the user choses any number other than 1 or 2, he will abort the deletion process by invoking the otherwise clause of the "Do Case" statement. This will close the database, release all memory variables and store "Y" to the loop control variable "exit". "Do while exit = N" will now test false and control of the program will revert back to the Main module.

7. Update Module

This module enables the user to enter new monthly information into the database for any one of the twenty Marine Corps Installations. When this module is called by the Main module, the first "Do while" loop, "Do while repeat = Y" tests true, and this loop is entered. Immediately following is the "Do while exit ="N" loop. This also tests true and control is passed to the lower level "Do while choice > 20" loop. This loop will present the user with a list of all twenty Marine Corps

Reporting Installations. The user now selects the number corresponding to the Installation which contains the information that he is interested in. If the user enters an improper value (i.e.,a letter or a number less than 1 or greater than 20), the "otherwise" portion of the case statement is executed, "choice" is set equal to 99 and an error message is displayed. The "Do while Choice > 20" loop tests true and the user is again presented with a menu of the Marine Corps Reporting Installations. This time he (hopefully!) picks an appropriate value and the case statement stores the choice to the memory variables "DBF" and "Station". DBF will be used in the calling module to access the appropriate database file. Station is used to ensure the appropriate title is used on all screens used in the calling module.

After the Station has been selected by the user, the program exits the "Do while Choice > 20" loop and executes the "Do while (month < 1) or (month > 12)" loop. This loop obtains the date of the desired report. The input is checked by the "If (month < 1) or (month > 20)" statement to ensure a valid date (in terms of number of months) is entered. The program will loop in this section of code until a valid entry is made. Once this is accomplished the date is stored in a memory variable called "Date". This memory variable will be part of the Key, used by calling program, to search for the appropriate record.

Once we have all of the necessary information (Station and Date) the program executes the "If verify = N" statement. This shows the user the values of the information which will be used to search the database for the record he has specified. If the user notes an error and enters anything other than "Y", "N" will stored to the memory variable "exit" and "Do while Exit = N" will test true. This will cause the program to loop to the beginning of this module and begin the whole selection process again. If all of the information is deemed correct, the program

stores "Y" to the memory variable "exit". This causes the "Do while exit = N" loop to test false and the "USE &DBF" statement is executed. This statement causes the program to access the database file of the installation which was chosen by the user. Next, the program moves the pointer to the top of the database file and initializes the loop control variable "count" to zero.

The next line executed, "Do while count < 28" also test true because we just set the value of count to zero. As a result we now increment our loop variable by one and execute the "Do Case" statement. The value of "count" determines which case statement is executed and hence what value the Crime memory variable will have. The program then moves on to initialize several temporary memory variables (TF2 trough TF28) and builds an input screen. The user now fills in the information. This information is stored by the program into the memory variables. Once all the fields have been filled in by the user, he is asked to verify that the information is correct. If the information is correct, the user enters a "Y" and the "If verify = Y" statement is executed. A blank record is appended onto the current database file and the values of the memory variables are loaded into the fields of this new record. If the information is not correct the "Else" portion of the "If verify = Y" statement is executed. This decrements the value of the variable "count" and stores "N" to the memory variable "exit". The program loops back to the top of the "Do While count < 28" loop and the user is allowed to reenter the correct information. The program will remain in this loop until it has appended a new record for each type of crime reported on. After all twenty seven records have been appended, the user is asked if he would like to update the crime statistics for another installation. If he answers yes, "If ans = Y" is executed. The current database is closed, all the memory variables are released and "repeat" (the loop

control variable for the "Do while repeat = Y" loop) is set to "Y". This causes the program to return to the beginning of the "Update" module and execute all over again. If the user does not want to update another file, the "Else" portion of "IF ans = "Y" is executed. This causes the database to be closed, all memory variables to be released and "N" is stored to repeat. Now the "Do while repeat = Y" loop will test false and the program will exit the loop. Control then is returned to the calling (Main) module.

8. Reports Module

This module presents the user with a menu which delineates the types of reports available to him. The menu is implemented by a simple case statement embedded within a "Do while" loop. The program will remain in the loop unless the user enters a valid number. Once a valid entry is made the program exits the loop to the appropriate subordinate module.

9. Monthly Module

This program allows the user to view any monthly report for any Marine Corps installation. It also provides the user with a hard copy of the report if desired. It accomplishes these tasks in the following manner.

The first several lines of code initialize all of the memory variables which will be used by the program. Next, the program the "Do while exit = N" loop and executes the "Do while choice > 20" loop. This loop presents the user with a menu of Marine Corps Installations. If the user enters an improper value (i.e. a letter or a number less than 1 or greater than 20), the "otherwise" portion of the case statement is executed, choice is set equal to 99 and an error message is displayed. The "Do while Choice > 20" loop tests true and the user is again presented with a menu of the Marine Corps Reporting Installations. This time he (hopefully!) picks

an appropriate value and the case statement stores the choice to the memory variables "DBF" and "Station". DBF will be used in the calling module to access the appropriate database file. Station is used to ensure the appropriate title is used on all screens used in the calling module.

After the Station has been selected by the user, the program exits the "Do while Choice > 20" loop and executes the "Do while (month < 1) or (month > 12)" loop. This loop obtains the date of the desired report. The input is checked to ensure a valid date (in terms of number of months) is entered. The program will loop in this section of code until a valid entry is made. Once this is accomplished the date is stored in a memory variable called "Date". This memory variable will be part of the Key, used by calling program, to search for the appropriate record.

Once an appropriate date has been entered, the program exits the "Do while Month" loop and executes the "If verify = N" statement. This shows the user the values of the information which will be used to search the database for the record he has specified. If the user notes an error and enters anything other than "Y", "N" will stored to the memory variable "exit" and "Do while Exit = N" will test true. This will cause the program to loop to the beginning of this module and begin the whole selection process again. If all of the information is deemed correct, the program stores "Y" to the memory variable "exit". This causes the "Do while exit = N" loop to test false the next statement, "USE &DBF", is executed.

"Use &DBF" is a macro substitution. It tells DBASE to open the database file indicated by the value of the DBF memory variable. The "Go Top" command, which follows the macro substitution, positions the record pointer to the top of the database file. Next the "Do While not EOF ()" command is executed. The function of this loop is to search the database for every occurrence of a crime that occurred

in the month which was specified by the "Date" memory variable. The "If substr(date 4,2) = substr(&date 4,2)" command checks each record for a date match. If one is found, the Case Statement, which is embedded in the "If substr (date, 4,2)..." clause, will assign the value of the crime field to an appropriate memory variable. Once the entire database file has been searched, we will have found all the occurrences for each of the twenty seven crimes during a specific month. For example, we now know how many Murders, Rapes, Assaults, etc. occurred in December, at MCAS, New River.

The purpose of the next series of "store" commands is simply to total each category of crimes. For example, Burglary included the Crimes of Entry, Attempted Entry, Attempted Entry of DOD property and Attempted Entry of Non-DOD property. Each one of these individual crimes is stored in the memory variable "Burglary" since it is the summation of all of those crimes. This is done for each reporting category of crimes. Following the store commands are a series of "@ say" commands. They format the results of the several store commands and present them on the screen. At this point the user is asked if he needs a hardcopy of the results. If he enters a "Y", we clear the screen and prompt him to prepare the printer. Next the program redirects output to the printer via the "set device to print command". When "eject" is executed, a carriage return and line feed are sent to the printer. The output is returned back to the screen by the "set device to screen command". After the report has been printed, we close the database, clear all memory variables and return control back to the Reports module. If the user did not want a hardcopy of the report, he would enter any letter other than "Y". This would cause the "IF ans <> Y" statement to test true and the "close all" and "clear

all" commands would be executed. Control would then be returned to the calling (Reports) module.

10. InfoAnnual Module

The purpose of this module is to make certain, before the user is allowed to compile an annual report, that a report does not already exist for the year in question. If a report has already been compiled for the year, the user is queried as to whether he wants to overwrite the information in the Archives database file. The module accomplishes these tasks in the following manner.

The first several lines of code build a screen and prompt the user to input the year of the Annual Report. The "If confirm = Y" statement, which is embedded within the "Do while repeat <> Y" loop, verifies that the correct year was entered. If it was not, the "If confirm = Y" statement tests false and the "Do while repeat <> Y" loop is executed once again. If the year was verified as being correct, the "If confirm = Y" statement tests true. This causes the program to access the Archives database file and search each record for a value of year which is equal to the value of year as entered by the user. This search is accomplished by the "Locate for year = curnt_year" statement. If no match is found (i.e.,an annual report does not exist) the "If not found () and EOF ()" condition tests true. The program then executes the "close all", "clear all" and "Do Annual" commands. This turns control over to the Annual.Prg module which actually compiles the report. Once the report is compiled control reverts back to the InfoAnnual.Prg module. The program now exits the "Do While Repeat <> Y" loop (because we set repeat equal to Y when we confirmed the year) and control reverts back to the calling (Reports) module.

If, however, a match had been found (i.e.,an annual report already exists for that year) the "If not found () and EOF()" statement would have tested false. In

this case, the "Else" portion of the If statement would be executed. At this point, the user would be asked if he wanted to overwrite the existing report. If he entered a "Y" the existing report would be deleted, the database packed and closed, and control of program execution would go to the Annual module. From this point on program execution would proceed as discussed above. If he chose not to overwrite the existing report, the "If continue = Y" statement would test false. At this point the program would exit the "Do while repeat <> Y" loop and return control of program execution to the calling (Report) module.

ll. **Annual Module**

This module searches every database file for crime statistics for the year chosen by the user. The yearly totals for each separate Installation are appended to the Archives database file utilizing a compound key (Station-Year) to uniquely identify each record.

The first several commands initialize all of the memory variables which will be used by the Annual.Prg module. Immediately following the initialization of the variables, we open the Archives database file in work area "A". Next we execute the "Do while choice < 21" loop. The purpose of this loop is to ensure that every installation database file is accessed when compiling the yearly totals. Note that within this loop, a case statement will assign a database file. The installation database file will be chosen based on the value of the loop control variable "choice" which is incremented after the yearly totals have been compiled for that specific installation. Once an installation database file has been selected, it is necessary to search for all occurrences of each type of crime for a given year. This is accomplished by the "If substr(date,4,2) = substr(&Date,4,2)" statement and the case statements which are embedded within the "Do while not EOF ()" loop. The

"Do while" loop will cause the record pointer to be advanced through each record in the database file. Each record is checked for a match on the current year by the "IF" clause. When a match is found, the number of occurrences for that crime is stored to a memory variable by the case statement. The program will loop through this sequence and continue to sum the occurrences of each separate crime type until end of file (EOF) is true. Note that using this embedded construction allowed us to locate each occurrence of a specific crime in a specific year and sum these values - all in one pass through the database file. Once we have the totals for each specific type of crime, we add the various memory variables together to arrive at values for different categories of crime. For instance, Homicides include Murder and Non-Negligent Manslaughter so we execute the command "Store Murdercount and Mancount to Homicides" to obtain this figure. Other totals are generated in an identical fashion.

After the totals have been compiled using the memory variables and the "store" command, we again open work area "A" (which holds the Archives database file) and append a new record. The "replace" commands will fill the blank fields of the newly appended record with the values held by the memory variables. After this is done, we increment the value of "count" (loop control variable for "Do while choice < 21") and the process starts over with a new Installation database file. The entire process repeats itself twenty times; the total number of Installations in the system.

12. Print Module

This is the main print module of the program. It prompts the user for the year of the annual report which is to be printed, and confirms the users input. It will loop in the "Do while repeat <> Y" loop until the user confirms that he has

entered the year that he wants. Once the year entered is confirmed by the user, the program will open the Archives database file and begin a search of the database for a record with the year indicated. If none is found, a message tells the user that a report for that year has not yet been compiled. The program then exits the Print module, closes the Archives database and returns to the Report module. If a record is found which matches the year required by the user, the program will execute each of the other subordinate print modules (Printl, Print2, Print3, Print4). After the last subordinate print module returns control to the main print module, the program exits the "Do while loop". Control now passes from the main print module back to the Reports module.

13. Printl Module

The purpose of this module is to print specific crime totals for each installation in a two column format. It will print the title of the report at the top of the page, list each installation and show current year and previous year totals. It also figures a grand total for each year as well as calculating percent change between the years.

This module is invoked by the main print module which opens the Archives database file and passes the value of curnt_year and last_year to Printl.Prg. Once invoked, the Printl.Prg module will clear the screen and display the message "Printing....". The output device is changed to the printer and the program executes the "Do while count < 15" loop. This loop uses a counter to determine which case statement is executed. The first collection of case statements is used to assign a title and a crime to the memory variables "Heading" and "Crime" respectively. Next, the program uses the "Heading" memory variable to compute the column position where the title will be printed. This ensures that for each of the

reports generated, each different title will be centered on the page. The next "Do while" loop, "Do while place < 21" will select the Station which we will use to search the Archives database file. This loop generates the text of the report as follows.

After we select a station, we use "Station" and "Last_year" to locate a record within the Archives database file. Once the record is located, we print the station name, and the number of crimes which occurred in that year (last_year). Next we reposition the pointer to the top of the file and search the Archives database file for a record which matches the Station and Curnt_year. The number of crimes which occurred in the current year is also printed on the same line as was the station and last year. The values of curnt_year and last_year are used to compute the percent change between the years. Once calculated, the percent change is printed on the same line as was the other information. Next, the program increments the current line by two and repeats the process for the next station in the case statement. In this manner, the totals for a specific crime (as indicated by the heading on the top of the page) are computed for every installation. Each line will contain then, the station name, last years total, current years total and the percent change between the years.

After all twenty installations have been processed the program will exit the "Do while place < 21" loop. It will now calculate a grand total for all of the installations for both years as well as calculating the percent change between the totals. This information is printed on the last line of the report. After the last line is printed, the "count" memory variable is incremented by one and the "Do while count < 15" loop is executed once again. The entire process is repeated again only now we are totaling statistics for the next crime in the first case statement. After all

the reports have been generated, (14 pages) the "Do while count < 15" loop tests true and the program exits the Printl module. Control of program execution is now returned to the Reports module.

14. Print2 Module

The purpose of this module is to print two reports in a column format. These are the DWI and DUI statistics for all of the Marine Corps installations. The module will put the title of the report at the top of each page, list each installation and show current year and previous year totals. It also figures a grand total for each year as well as calculating the percent change between the years.

The functioning of this module is identical to that of Printl.Prg with the exception that it formats into eight separate columns and produces only two individuals reports.

The "Do while place < 21" loop serves the same purpose as discussed in the Printl.Prg module. Once a specific crime is chosen, DWI for example, it will access the Archives database and get the information pertaining to that crime for the previous year and current year. This information is presented by Installation, on each separate line of the report. After all the information for all of the installations has been printed, the count memory variable of the "Do while count < 3" loop is incremented and the process repeats itself. This time around we search the database for DUI statistics. Once these are all compiled, the "count" memory variable is incremented again. This time "Do while count < 3" tests false and we exit the loop. Control is now passed back to the calling (Reports.Prg) module.

15. Print3 Module

The purpose of this module is to produce four tables which will show the statistics for the present reporting categories of crimes. The first table, "Index

Crimes Reported to HQMC" presents the current year and previous year totals for the crimes of Homicide, Forcible Rape, Robbery, Aggravated Assault, Burglary, Larceny and Motor Vehicle Theft. It also lists the rates at which these crimes occurred (occurrence/1000 individuals) and the totals for the year. The second and Third tables present the same information but as two separate tables. The table labeled "Violent Crimes Reported to HQMC" just shows the statistics for Homicide, Rape, Robbery and Aggravated Assault. The table titled "Crimes Against Property Reported to HQMC" shows the statistics for Burglary, Larceny and Motor Vehicle Theft. The last table generated, "Narcotics/Dangerous Drugs and Marijuana Offenses Reported to HQMC", delineates the drug offense (sales, possession, etc), totals and the rates of occurrences.

The program generates the four reports as follows. The "Sum" function is used to search the Archives database file and sum the values of the specified fields into temporary memory variables. This is done for both the current year and the previous year. The current year values are stored in the T_Homicide, T_Rape etc. memory variable and the previous years values are stored as L_Homicide, L_Rape etc. After all of the values have been summed and stored into the appropriate memory variables, the "Do while counter < 12" loop is executed. It is within this loop that the rates of occurrences and yearly totals for each specific crime are computed. For example, the very first time the loop is executed, the value of counter is one. The case statement which is embedded within the "Do while" loop assigns the value of T_Homicide to memory variable "Crime1" and the value of L_Homicide is stored to "Crime2". Additionally, "Percntchg1" is stored to Percntchg, "CT_Rate1" is stored to "CT_Rate" and "LST_Rate1" is stored to "LST_Rate". We next use these memory variables to calculate the percentage

change between years and the rate of occurrence for each year. This is accomplished by the three "Store" statements immediately following the case statement. Note that we use the macro substitution (&) to keep track of each statistic as we loop through this section of code. In other words, when we compute the percent change for "Homicide" we store this value to "Prcntchg1". When we calculate the percent change for Rape we would store that value to "Prcntchg2" and so forth until all eleven case statements are executed. At this point we would exit the "Do while counter < 12" loop and enter the first print sequence. Incidentally, the "If counter < 8" statement serves to keep a running total of the seven Index Crimes. The Drug Offenses (case statements 9 through 11) are not included in this total and the "If" statement prevents their inclusion.

The first print sequence starts with the command "@ 0,0" and ends with the "eject" command. All of the code in between these two commands simply instructs the printer to format the output in the manner desired by the user.

When the eject command is executed, it causes the printer to advance to the top of the next page. At this point, the program begins to execute the second print sequence. This sequence generates a report of the "Violent Crimes Reported to HQMC". Again, we use a series of "store" commands to calculate the totals for this report. Once these statistics are calculated, all of the memory variables are formatted as output by use of the "@, say" constructions. At the end of the print sequence, another "eject" command is executed and a new page scrolls onto the printer. The same process occurs for the last two reports which are generated. After the last "eject" command is executed, we issue the "Set device to screen" command to redirect output to the screen. Control of program execution is then passed back to the calling (Print.Prg) module.

16. Print4 Module

This module will calculate the crime rate per 1000 individuals for each of the major crime reporting categories. It will do this for every reporting installation and will present the results in a bar graph format.

The first few lines of code simply clear the screen and generate a message to the user. The output is next routed to the printer via the "Set device to print" command. The program next enters the "Do while count < 5" loop. This loop causes the program to generate the four separate reports. The first time it is executed the title "Index Crime Rate by Base (per 1000) " will be stored to the memory variable "Heading" and "Indextotal" will be stored to the memory variable "Crime". The program will next calculate the center of the page and print the heading centered on the top of the page. The "Do while Place < 21" loop is executed next. The program will loop through this sequence of code obtaining the indextotals for each installation and printing the bar graph for that installation. It does this by first using the "locate" command to find correct record. Once that record is located, we check the value of the population field. If it is zero, we print an error message. If it is some value greater than zero, it is used to compute the crime rate for that installation, for that year. The "Do while start < Decrement" loop is the portion of code which generates the right amount of characters in our bar graph. After that line of the bar graph is printed, the program enters the second "locate" sequence. This portion of code is identical to that of the first locate sequence with the exception of the year which is used to locate the appropriate record. Where the first locate clause searched for all instances of indextotal in say, MCLB Albany in 1985, this locate command will search for all instances of indextotal in MCLB Albany in 1986. The bar graph for that year will be generated

by the second "Do while start < decrement" loop in the same manner as the first was.

After the two bar graphs are printed for the first installation, the program will increment the line value and the value of "Place". This will cause the program to repeat the entire "Do while Place" loop for the next installation in the case statement. After the program has gone through all of the installations, the value of Place will be twenty one and the program will exit the "Do while place < 21" loop. The "If count > 3" statement which is executed next, will print the appropriate scale on the bottom of the report. At this point the program will loop to the beginning of the "Do while count < 5" loop and begin to print the next report in the sequence. After all four reports have been generated the program exits the "Do while count < 5" loop, routes output back to the screen, releases all memory variables and returns control of program execution to the calling (Print.Prg) module.

VI. CONCLUSIONS

This thesis proposed a Database Management System suitable for implementation within the Military Police Section of the Operations Division, Plans, Policies, and Operations Department, Headquarters Marine Corps. This system could also be utilized by the individual reporting installations with only slight modification.

The goal of this study was to develop a system which would allow for faster information retrieval, with better accuracy at reduced cost (in terms of man hours). This system could also be used to aid local commanders in identifying crime trends and in implementing crime prevention measures which address specific criminal activity in their area.

By utilizing the Life Cycle approach to systems analysis and design, we were able to build a system which was tailored to the unique needs of the Military Police Section. Because of its emphasis on documentation and maintenance considerations, this system will be able to be expanded to meet changing user requirements.

The following recommendations are offered as possible enhancements to the current system. Because the program requires a great deal of disk accesses a hard disk is recommended. The search, seek and data transfer raters are all much more rapid utilizing Winchester drives as opposed to floppy disk drives.

The use of a DBase compiler would also enhance the program. Compiled programs execute several orders of magnitude faster than do interpreted programs. There are many commercially available packages which will compile and optimize DBase Code. (Clipper and dBIII)

The military Police Section should conduct a study to determine what their actual reporting requirements are. As was discussed in chapter four of the thesis, a good deal of the information which is collected is not used for reporting purposes. Many of the logical relationships that exist between the entities of the enterprise are not used. It would be more beneficial in terms of data integrity, to delete columns three thru 28 on NAVMC 1630/1 rather than to carry information which is not used or useful.

LIST OF REFERENCES

1. Davis, Gordon B. and Olson, Margreth H., Management Information Systems, Conceptual Foundations, Structure and Development, McGraw-Hill Series in Management Information Systems, 1985.

2. Davis, William S., Systems Analysis and Design, Addison Wesley, 1983.

3. Meilir, Page-Jones, The Practical Guide to Structured Systems Design, Yourdon Press, 1980.

4. Everst, Gordon, Database Management, McGraw-Hill, 1986.

5. DeMarco, Thomas, Structured Analysis and System Specification, Yourdon Press, 1978.

6. Kroenke, David, Database Processing, Science Research Associates, 1983.

7. Walters, Richard F., Database Principles for Personal Computers, Prentice Hall, 1987.

8. Kent, Willaim A., A Simple Guide to Five Normal Forms in Relational Database Theroy, Communications of the ACM, February, 1986.

9. Information Technology Services, Microcomputer Database Management Systems, Their Selection and Evaluation, Stanford University, 1984.

10. Pressmen, Roger S. , Software Engineering: A Practitioner's Approach, McGraw-Hill Series in Software Engineering and Technology, 1987.

APPENDIX A

Appendix A contains the Data Dictionary. The definitions of all representations on the various Data Flow Diagrams are contained here. Each piece of composite data (i. e.,Data Flows, Data Files, and Data Elements) is defined in terms of the "atomic" elements of which it is composed.

In order to facilitate the use of the Data Dictionary the following notation is used:

=	means "is composed of"
+	means "and"
()	indicates the data element is at the atomic level. Its structure and type can be found in the "Data Base File Structures" listed in Appendix C
*	indicates clarifying comments can be found where indicated
{ }	indicates iteration of 1 to n of the element

DATA DICTIONARY

DATA FLOWS

Access Info =	Log-in Info + Logdate + Logtime + Logchoice
Ad Hoc Reports =	* all elements user defined*
Audit Report =	Log-in Info + Access Info
Crime Information =	NAVPERS 1630/1 + NAVMC 11007 + NAVMC 11007 ADD.
Crime Statistics Report =	Station + Homicide + Forcerape + Robbery + Assault + Burglary + Larceny + Autotheft + Narccount + Sellcount + Dealcount + Smokecount + DWI + DWIBase + DUI + DUIBase + Indextotal + Violent + Propcrime + Drugtotal + DUItotal + DWItotal + Boozetotal + Year + Population
Installation Name =	(character 8 bytes)
Log-in Info =	Loguser + Loguit + Logrank
Monthly Report =	Station + Date + Indextotal + Narccount + Sellcount + Smokecount + Dealcount + Boozetotal
NAVMC 11007 =	DUIBase + DUI + DWIBase + DWI + Station + Date
NAVMC 11007 ADD. =	Station + Date

DATA DICTIONARY

NAVPERS 1630/1 = Homicide + Forcerape + Robbery + Assualt + Burglary + Larceny + Autotheft + Drugtotal + Station + Date + F-1 + F-2 + F-3 + F-4 + F-5 + F-6 + F-7 + F-8 + F-9 + F-10 + F-11 + F-12 + F-13 + F-14 + F-15 + F-16 + F-17 + F-18 + F-19 + F-20 + F-21 + F-22 + F-23 + F-24 + F-25 + F-26 + F-27 + F-28

Data Files

Access File = {Access Info}

Archives File = { Crime Statistics Report }

Crime File = { Installation Reports }

Installation File = {Station}

DATA ELEMENTS

Arapecount = *alias attempted rape * *see item 2B Figure 5.2*

Armedcount = *alias robbery w/firearm * *see item 3a Figure 5.2*

Assault = Assltcount + Othercount + Simplcount

Assltcount = *alias assault w/firearm* *see item 4a Figure 5.2*

DATA DICTIONARY

Autocount =	*alias motor vehical theft - auto* *see item 7a Figure 5.2*
Autotheft =	Autocount + Vancount + Gautocount + Nautocount
Boozetotal =	DWItotal + DUItotal *alias alcoholtotal
Burglary =	Entrycount + Trycount + DODcount + NDODcount
Civiltheft =	*alias larceny non-govt. property* *see item 6b Figure 5.2*
Crime =	Murdercount + Mancount + Rapecount + Arapecount + Armedcount + Othercount + Assltcount + Otwpncount + Simplcount + Entrycount + Trycount + DODcount + NDODcount + Govtheft + Civiltheft + Autocount + Vancount + Nautocount + Gautocount + Narccount + Sellcount + Dealcount +Smokecount +DWIBase + DWI + DUIBase +DUI
Date =	[year/month]
Dealcount =	*alias marijuana sale & trafficing* *see item 9B Figure 5.2*
DODcount =	*alias bugrlary & housebreaking govt. property* *see item 5c Figure 5.2*

DATA DICTIONARY

Drugtotal =	Narccount +Sellcount + Dealcount + Smokecount
DUI =	*alias DUI off base*
DUIbase =	*alias DUI on base*
DUItotal =	DUI + DUIbase
DWI =	*alias DWI off base*
DWIbase =	*alias DWI on base*
DWItotal =	DWI + DWIbase
Edited Crime Information =	*Crime Information*
Entrycount =	*alias burglary & housebreaking* *see item 5a Figure5.2*
F-1.....F-28 =	*alias of fields in Figure 5.2*
ForceRape =	Rapecount + Arapecount
Gautocount =	*alias motor vehical theft - govt.* *see item 7c Figure 5.2*
Govtheft =	*alias larceny of govt. property* *see item 6a Figure 5.2*
Homicide =	Murder + Mancount

DATA DICTIONARY

Indextotal =	Homicide + Forcerape + Robbery + Assault + Burglary + Larceny + Autotheft
Larceny =	Govtheft + Civiltheft
Logchoice =	(character 10 bytes)
Logdate =	(date 8 bytes)
Logrank =	(character 5 bytes)
Logtime =	(character 8 bytes)
Logunit =	(character 4 bytes)
Loguser =	(character 20 bytes)
Mancount =	*alias manslaughter* *see item 1b Figure 5.2*
Murdercnt =	*alias murder* *see item 1a Figure 5.2*
Narccount =	*alias Narcotics use & possession* *see item 8A Figure 5.2*
Nautocount =	*alias motor vehical theft non-govt. property* *see item 7d Figure 5.2*
NDODcount =	*alias burglary & housebreaking non-govt. property* *see item 5d Figure 5.2*

DATA DICTIONARY

Othercount =	*alias robbery w/other weapon * *see item 3b Figure 5.2*
Otwpncount =	*alais assault w/other weapon* *see item 4b Figure 5.2*
Population =	(character 5 bytes)
Propcrime =	Burglary + Larceny +Autotheft
Rapecount =	*alias Rape* *see item 2A Figure 5.2*
Robbery =	Armedcount + Othercount
Sellcount =	*alias narcotics sales &trafficing* *see item 8B Figure 5.2*
Simplcount =	*alias simple assault* *see item 4c Figure 5.2*
Smokecount =	*alias marijuana use & possession* *see item 9A Figure 5.2*
Station =	(character 30 bytes)
Trycount =	*alias burglary & house breaking entry* *see item 5b Figure 5.2*
Vancount =	*alias motor vehical theft - other* *see item 7b Figure 5.2*
Violent =	Homicide + Forcerape +Robbery + Assault

DATA DICTIONARY

Year = (numeric 4 bytes)

Yearly totals = { installation reports}

APPENDIX B

MiniSpecs

Process Description Summary Page

Process Name: Security

Process 1.0

```
Do while incorrect password is entered
    get the user's name
    get the user's rank
    get the user's unit
    get the user's password
    if after three tries they do not enter the correct password
        exit to the operating system
    if the correct password is entered
        do the main program
Enddo while password is incorrect
```

PROGRAMMER NOTES: Use global variables to store the user's name, rank and unit after the password has been correctly entered

Process Description Summary Page

Process Name: Log-In

Process 2.0

After the user enters the system do

record the user's name in the access database

record the user's rank in the access database

record the user's unit in the access database

record the date of access

record the time of access

record the menu option chosen

PROGRAMMER NOTES: When this process is coded, the modules functioning should be transparent to the user. Writing to the file should take place while the user is viewing the main menu options.

Process Name: Update Files

Process 3.0:

Get the date of the desired report

Get the installation name for that report

Get the type of crime statistic needed for the report

Store this information in memory

PROGRAMMER NOTES: This process obtains the necessary information needed to search for a specific record within a specific database. A series of menu screens will aid the user in entering the necessary information.

Process Description Summary Page

Process Name: AddInfo

Process 3.2

Do the following for all installations
get installation name
get date of report
verify choices
Do for all categories of crime
get type of crime
enter crime data
get next crime
enddo for all crime
Enddo for all installations

PROGRAMMER NOTES: This process obtains the information necessary to add a new record or records to an installation database. Information by type of crime should be entered for all crime categories of crime within each separate database.

Process Name: EditInfo

Process 3.3

Use GetInfo process to locate specific record
display record on screen
allow user to edit record
verify any changes made
record changes in appropriate database

PROGRAMMER NOTES: Use memory variables to manipulate information stored in the database. Do not allow the user to edit information directly into the database.

Process Name: Delete Info

Process 3.4

Use GetInfo process to locate a specific record
display record on screen
ensure that user wants to delete that record
delete record

PROGRAMMER NOTES: Ensure database is packed after the record has been deleted.

Process Description Summary Page

Process Name: Generate Crime Statistics Report

Process 4.1

Do for all installations
    select installation database
    get population figure from the user
    do for all categories of crime
        store crimes to memory variables
        append blank record to database file
        replace empty fields with memory variables
Enddo

PROGRAMMER NOTES: This process should search through all of the database files for records having the same value in the year field. Each crime statistic for that specific year is totaled and the results will be stored in the Crime Statistics Report database file.

Process Name: Ad Hoc Reports

Process 4.2

PROGRAMMER NOTES: This process will be implemented by the Data Manipulation Language (DML) of the application used to implement the system.

Process Name: Monthly Reports

Process 4.3

get date of required report
get installation name
use installation database
    search for all record having the required date
    display all information
    print information if desired

PROGRAMMER NOTES: This process will provide a synopsis of the major crime categories which occurred at a specific installation during a given month. This information will be extracted from the Archive Reports database file

Process Description Summary Page

Process Name: Audit Reports

Process 4.4

use Access database file
display fields on screen
ask if printout is required
if required, print report
else return to reports menu

PROGRAMMER NOTES: This report should be a simple listing of who accessed the system, the date and time of access, as well as the main menu options chosen by the user

APPENDIX C

DATA BASE STRUCTURE
ACCESS.DBF

FIELD	FIELD NAME	TYPE	WIDTH
1	Loguser	Character	20 Bytes
2	Logunit	Character	4 Bytes
3	Logchoicec	Character	10 Bytes
4	Logchoice	Numeric	2 Bytes
5	LogTime	Character	8 Bytes
6	Logdate	Date	8 Bytes

Programmers note: This file provides an audit trail of accesses to the system. The file is restricted to a read only mode for the user. The program will update the file each time a user logs on to the system.

DATA BASE STRUCTURE ARCHIVES.DBF			
FIELD	**FIELD NAME**	**TYPE**	**WIDTH**
1	Station	Character	30 Bytes
2	Homicides	Numeric	4 Bytes
3	Forcerape	Numeric	4 Bytes
4	Robbery	Numeric	4 Bytes
5	Assault	Numeric	4 Bytes
6	Burglary	Numeric	4 Bytes
7	Larceny	Numeric	4 Bytes
8	Autotheft	Numeric	4 Bytes
9	Narccount	Numeric	4 Bytes
10	Sellcount	Numeric	4 Bytes
11	Smokecount	Numeric	4 Bytes
12	Dealcount	Numeric	4 Bytes
13	DWIBase	Numeric	4 Bytes
14	DWI	Numeric	4 Bytes
15	DUIBase	Numeric	4 Bytes
16	DUI	Numeric	4 Bytes
17	Indextotal	Numeric	4 Bytes
18	Violent	Numeric	4 Bytes
19	Propcrime	Numeric	4 Bytes
20	Drugtotal	Numeric	4 Bytes
21	TotalDUI	Numeric	4 Bytes
22	TotalDWI	Numeric	4 Bytes
23	Boozetotal	Numeric	4 Bytes
24	Year	Character	5 Bytes
25	Population	Numeric	5 Bytes

Programmers note; also see Figure 5.3 labled "Archives Database Structure"

DATA BASE STRUCTURE
INSTALLATION .DBF FILES

FIELD	FIELD NAME	TYPE	BYTES
F-1	offenses on base	Numeric	2
F-2	offenses of base	Numeric	2
F-3	unfounded false reports on base	Numeric	2
F-4	unfounded false reports off base	Numeric	2
F-5	cleared by arrest	Numeric	2
F-6	investigated by NIS	Numeric	2
F-7	rate per 1000 personnel assigned	Numeric	2
F-8	total this month last year	Numeric	2
F-9	perpetrator - USMC	Numeric	2
F-10	perpetrator - other services	Numeric	2
F-11	perpetrator - dependent	Numeric	2
F-12	perpetrator - DOD personnel	Numeric	2
F-13	perpetrator - male	Numeric	2
F-14	perpetrator - female	Numeric	2
F-15	perpetrator - Caucasian	Numeric	2
F-16	perpetrator - Negroid	Numeric	2
F-17	perpetrator - all others	Numeric	2
F-18	victim - USMC	Numeric	2
F-19	victim - other services	Numeric	2
F-20	victim - dependent	Numeric	2
F-21	victim - DOD personnel	Numeric	2
F-22	victim - male	Numeric	2
F-23	victim - female	Numeric	2
F-24	victim - Caucasian	Numeric	2
F-25	victim - Negroid	Numeric	2
F-26	victim - all others	Numeric	2
F-27	drug involvement	Numeric	2
F-28	alcohol involvement	Numeric	2
CRIME	classification of offense	Character	30
DATE	date of offense	Character	5

Programmers note; a descriptive summary of the data fields is provided in Figure 4.9

APPENDIX D

The following pages present the module descriptions for the 19 individual modules supporting the application. The descriptions presented are in the form of pseudocode and do not represent the actual code found in the DBase III+ application If the reader requires that level of knowledge he or she is directed to Appendix E where a complete listing of the program code itself is provided.

Module Description Summary Page (1 of 2)

MODULE NAME: Annual.Prg CALLED FROM: Infoannual.Prg	DATA BASES USED: Archives.DBF (and all .DBF Files)

THIS MODULE CALLS: none

```
STORE 0 TO ALL MEMORY VARIABLES
SELECT A
USE ARCHIVE
DO WHILE CHOICE < 21
    DO CASE
        SELECTS EACH DATABASE IN SEQUENCE
    END CASE
    DO WHILE ANSWER = N
        GET POPULATION
        IF REPLY = Y
            CONFIRM RESPONSE
        ELSE
            GET POPULALTION
        ENDIF
    ENDDO WHILE ANSWER = N
    USE &DBF
    DO WHILE .NOT. EOF()
        IF SUBSTDR (DATE 4,2) = SUBSTR (&DATE,4,2)
            LOCATES RECORD BY DATE MATCH (YEAT)
            DO CASE
                    TOTALS VALUES FOR EACH CRIME CATEGORY
            ENDCASE
        ELSE
            ERROR CONDITION - ON DATE
        ENDIF
        SKIP
    ENDDO WHILE .NOT. EOF()
    STORE MURDERCNT + MANCOUNT TO HOMICIDES
    TOTALS COMPOSIT INDEX CRIMES
    IF VAL (SUBSTR ("& DATE",4,2)) > (80)
        PREPEND PROPER CENTRY TO DATE
    ELSE
        PREPEND PROPER CENTURY TO DATE
    ENDIF
    TINDXTOTAL = TINDXTOTAL + INDXTOTAL
    TOTALS INDEX CRIMES/STATION
```

Module Description Summary Page (2 of 2)

MODULE NAME: Annual.Prg CALLED FROM: Infoannual.Prg	DATA BASES USED: Archives.DBF (and all .DBF Files)

THIS MODULE CALLS: none

```
        SELECT A
        APPEND BLANK
        APPENDS NEW RECORDS TO ARCHIVE DBF
        REPLACE STATION WITH "&STATION"
        FILLS BLANK FIELDS  IN APPENDED RECORD
        STORE POP + LTRIM (STRING (CHOICE,2)) TO TEMP
        BUILDS POP MEMORY VARIABLE FOR EACH STATION
        REPLACE POPULATION WITH &TEMP
        FILLS POPULATION FIELD IN ARCHIVE RECORD
        TINDXCH = CHOICE + 1
        RELEASE ALL EXCEPT TINDX*
        CHOICE = TINDXCH
    ENDDO WHILE CHOICE < 21
    CLOSE ALL
    CLEAR ALL
    RETURN
```

PROGRAMMER NOTES: This module searches each database for crime statistics for the year chosen by the user. The user is prompted for the population of each installation as that database is accessed. All of the information is collated and appended to the Archives database file.

Module Description Summary Page

MODULE NAME: Delete.Prg
CALLED FROM: Main.Prg

DATA BASES USED: As selected by user
(any .DBF File)

THIS MODULE CALLS: GetInfo.Prg

```
STORE N TO EXIT
DO WHILE EXIT = N
        DO GETINFO
        *
        USE &DBF
        LOCATE FOR CRIME = &CRIME) .AND (DATE = &DATE)
        IF .NOT. FOUND  .AND. EOF()
            ASK FOR VERIFICATION OF INPUT
            CLOSE ALL
            CLEAR ALL
        ELSE
            DO WHILE REPEAT = Y
                STORE Y TO VERIFY
                *
                IF VERIFY = Y
                    STORE Y TO EXIT
                    STORE N TO REPEAT
                    REPLACE FIELD VALUES W/ MEMORY VARIABLES
                    *
                ELSE
                    STORE Y TO REPEAT
                ENDIF VERIFY
            ENDDO WHILE REPEAT
        ENDIF .NOT. FOUND .AND. EOF()
ENDDO WHILE EXIT = N
CLOSE ALL
CLEAR ALL
RETURN
```

PROGRAMMER NOTES: This module allows the user to delete erroneous or redundant records.

Module Description Summary Page

MODULE NAME: Edit.Prg CALLED FROM: Main.Prg	DATA BASES USED: As selected by user (any .DBF File)

THIS MODULE CALLS: GetInfo.Prg

```
STORE N TO EXIT
DO WHILE EXIT = N
        DO GETINFO
        *
        USE &DBF
        LOCATE FOR CRIME = &CRIME) .AND (DATE = &DATE)
        IF .NOT. FOUND  .AND. EOF()
            ASK FOR VERIFICATION OF INPUT
            CLOSE ALL
            CLEAR ALL
        ELSE
            DO WHILE REPEAT = Y
                STORE Y TO VERIFY
                *
                IF VERIFY = Y
                    STORE Y TO EXIT
                    STORE N TO REPEAT
                    REPLACE FIELD VALUES W/ MEMORY VARIABLES
                    *
                ELSE
                    STORE Y TO REPEAT
                ENDIF VERIFY
            ENDDO WHILE REPEAT
        ENDIF .NOT. FOUND .AND. EOF()
ENDDO WHILE EXIT = N
CLOSE ALL
CLEAR ALL
RETURN
```

PROGRAMMER NOTES: This module allows the user to select a specific record for editing.

Module Description Summary Page

MODULE NAME: Infoannual.Prg CALLED FROM: Report.Prg	DATA BASES USED: Archives.DBF

THIS MODULE CALLS: Annual.Prg

```
STORE N TO CONTINUE, REPEAT, CONFIRM
(BUILDS SCREEN)
DO WHILE REPEAT <> Y
    GET YEAR OF DESIRED REPORT
    CONFIRM THE YEAR ENTERED
    IF CONFIRM = Y
        STORE Y TO REPEAT
        USE ARCHIVE
        INDEX ON STATION + STR(YEAR) TO TEMP
        LOCATE FOR YEAR = CURNT_YEAR
        IF NOT FOUND () AND EOF ()
            CLOSE ALL
            CLEAR ALL
            DO ANNUAL
        ELSE
            ISSUE WARNING MESSAGE
            ASK TO CONTINUE
            IF CONTINUE = Y
                DELETE ALL FOR YEAR = CURNT_YEAR
                PACK
                CLOSE ALL
                DO ANNUAL
            ELSE
                CLOSE ALL
            ENDIF
        ENDIF
    ENDIF
ENDDO WHILE REPEAT <> Y
RETURN
```

PROGRAMMER NOTES: This module obtains the year of the annual report and checks to see if a report already exists for that year. If a report exists, the user is given the option of over-writing the report or exiting to the Reports.Prg module.

Module Description Summary Page

MODULE NAME: Main.Prg
CALLED FROM: Security.Prg

DATA BASES USED: none

THIS MODULE CALLS: Init.Prg, Edit.Prg, Update.Prg, Delete.Prg, Reports.Prg

```
DO INIT
*
DO WHILE EXIT > 5
    *
    (SHOW MENU)
    *
    DO CASE
        *
        (SELECT OPTION)
    OTHERWISE
        *
        (ERROR MESSAGE)
        *
        DO WHILE DELAY < 50
            STORE DELAY + 1 TO DELAY
        ENDDO WHILE DELAY < 50
    ENDCASE
ENDDO WHILE EXIT > 5
RETURN
```

PROGRAMMER NOTES: This module presents the user with a menu of all the program reports available. It also controls the execution of all sub-programs.

Module Description Summary Page (1 of 2)

MODULE NAME: Monthly.Prg CALLED FROM: Reports.Prg	DATA BASES USED: As selected by user (any .DBF File)

THIS MODULE CALLS: GetInfo.Prg

```
STORE 0 TO ALL MEMORY VARIABLES
DO WHILE EXIT = N
    DO WHILE CHOICE > 20
        BUILD MENU SCREEN
        DO CASE
            GET APPROPRIATE DATABASE FILE
        OTHERWISE
            DISPLAY ERROR MESSAGE
        ENDCASE
    END DO WHILE CHOICE > 20
    DO WHILE (MONTH <1) OR (MONTH > 12)
            GET DATE
        IF (MONTH <1) OR (MONTH >12)
            PRINT ERROR MESSAGE
        ENDIF
    ENDDO WHILE (MONTH <1) OR (MONTH >12)
    GET VERIFICATION OF INPUT
    IF VERIFY = Y
        STORE Y TO EXIT
    ELSE
        STORE N TO EXIT
        STORE 99 TO CHOICE
    END IF
ENDDO WHILE EXIT = N
USE &DBF
GO TOP
DO WHILE NOT EOF ()
    IF SUBSTR (DATE,4,2) = SUBSTR ("&DATE,4,2)
        DO CASE
            STORE CRIME FIGURES TO MEMORY VARIABLES
        ENDCASE
    ENDIF
    SKIP
ENDDO WHILE NOT EOF()
```

Module Description Summary Page (2 of 2)

MODULE NAME: Monthly.Prg CALLED FROM: Reports.Prg	DATA BASES USED: As selected by user (any .DBF File)

THIS MODULE CALLS: GetInfo.Prg

```
STORE ALL MEMORY VARIABLES
TOTALS THE INDEX CRIMES
CLEAR SCREEN
(SHOW TOTALS ON SCREEN)
(ASK IF HARD COPY IS NEEDED)
IF ANS  <> Y
   CLOSE ALL
   CLEAR ALL
ELSE
   CLEAR SCREEN
   SET DEVICE TO PRINT
   EJECT (FORCE CARRAGE RETURN
   SET DEVICE TO SCREEN
   CLOSE ALL
   CLEAR ALL
ENDIF ANS <> Y
RETURN
```

PROGRAMMER NOTES: This module allows the user to view any monthly report which is stored in the database files. It also allows the user to get a hardcopy of the report if desired

Module Description Summary Page

MODULE NAME: Print.Prg CALLED FROM: Reports.Prg	DATA BASES USED: Archives.DBF

THIS MODULE CALLS: Print1.Prg, Print2.Prg, Print3.Prg, Print4.Prg

```
(BUILD SCREEN)
IF ANS <> Y
    RETURN
ELSE
    DO WHILE REPEAT <> Y
        *
        *
        GET YEAR OF REPORT
        *
        IF CONFIRM =Y
            *
            CONFIRM ENTRY
            OPEN DATABASE
            SEARCH FOR YEAR
            IF NOT FOUND () AND EOF()
                *
                GENERATE ERROR MESSAGE
            ELSE
                DO ALL PRINT ROUTINES
            END IF NOT FOUND
        ENDIF CONFIRM
    ENDDO WHILE REPEAT
ENDIF ANS
ERASE TEMP
CLOSE ALL
CLEAR ALL
RETURN
```

PROGRAMMER NOTES: This module controls all of the print routines contained in the program.

Module Description Summary Page

MODULE NAME: Print1.Prg CALLED FROM: Print.Prg	DATA BASES USED: Archives.DBF

THIS MODULE CALLS: none

```
DO WHILE COUNT < 15
    DO CASE
        SELECTS CRIME FOR DATABASE SEARCH
        STORES APPROPRIATE HEADING FOR PRINTING
    ENDCASE
    STORE (80-LEN(TRIM(&HEADING)))/2 TO CENTER
    COMPUTES CENTER OF PAPER FOR HEADING PLACEMENT
    PRINTS HEADING
    DO WHILE PLACE < 21
        DO CASE
            SELECTS STATION FOR DATA BASE SEARCH
        ENDCASE
        GO TOP
        LOCATE FOR (STATION = &STATION) .AND. (YEAR = LAST YEAR)
        FINDS SPECIFIED RECORD BASED ON COMPOUND KEY
        PRINTS STATION NAME, VALUE OF CRIME FIELD
        INCREMENTS TOTAL BY VALUE OF CRIME FIELD
        GO TOP
        LOCATE FOR (STATION =&STATION) .AND. (YEAR = CURRENT YEAR)
        FUNCTIONS AS ABOVE, BUT USES CURRENT YEAR FIGURES
        IF TEMP =0
            CHECK FOR DIVISION BY ZERO
        ELSE
            COMPUTES PERCENTAGE CHANGE BETWEEN YEARS
        ENDIF
        STORE PLACE +1 TO PLACE
         COMPUTES FIGURES FOR NEXT INSTALLATION IN DO WHILE LOOP
    ENDDO WHILE PLACE < 21
    PRINTS TOTAL ON LAST LINE
    IF YEAR TOTAL =0
        CHECK FOR DIVISION BY ZERO
    ELSE
        COMPUTES TOTAL PERCENTAGE CHARGE BETWEEN YEARS
    ENDIF
    INCREMENT COUNT TO GET INFO FOR NEXT CRIME
ENDDO WHILE COUNT < 15
RELEASE ALL
RETURN
```

PROGRAMMER NOTES: This module will print the crime statistics information in the two column format as well as calculate the percent change between the years

Module Description Summary Page

MODULE NAME: Print2.Prg CALLED FROM: Print.Prg	DATA BASES USED: Archives.DBF
THIS MODULE CALLS: none	

```
DO WHILE COUNT < 3
    DO CASE
        STORE HEADING
        STORE CRIME
    ENDCASE
    STORE (80-LEN(TRIM(&HEAQDING)))/2 TO CENTER
    PRINTS APPROPRIATE HEADING IN CENTER OF PAGE
    STORE 11 TO LINE
    STORE 1 TO PLACE
    STORE 0 TO ALL MEMORY VARIABLES
    DO WHILE PLACE < 21
        DO CASE
            SELECTS INSTALLATION FOR DATABASE SEARCH
        ENDCASE
        GO TOP
        LOCATE FOR (STATION+&STATION) .AND. (YEAR = &YEAR)
        FINDS SPECIFIED RECORD BASED ON COMPOUND KEY
        STORES FIELD VALUES TO MEMORY VARIABLES
        GO TOP
        LOCATE FOR (STATION = &STATION) .AND (YEAR = CURNT_YEAR)
        AS ABOVE BUT FOR THE CURRENT YEAR
        IF LAST _TOTAL =0
            CHECKS FOR DIVISION BY ZERO
        ELSE
            COMPUTES TOTAL PERCENT CHANGE
        ENDIF
        STORE PLACE + 1 TO PLACE
        STORE LINE + 2 TO LINE
        INCREMENTS DO WHILE AND CALCULATES AND PRINTS NEXT LINE
    ENDDO WHILE PLACE < 21
    STORE LINE + 1 TO LINE
    PRINTS TOTALS AT BOTTOM OF PAGE
    IF TEMP 5 TOTAL =0
        CHECK FOR DIVISION BY ZERO
    ELSE
        CALCULATE TOTAL PERCENT CHANGE
    ENDIF
    INCREMENTS COUNT FOR
    RELEASE ALL TEMPORARY MEMORY VARIABLES
ENDDO WHILE COUNT < 3
RETURN
```

Module Description Summary Page

MODULE NAME: Print3.Prg CALLED FROM: Print.Prg	DATA BASES USED: Archives.DBF

THIS MODULE CALLS: none

```
@10, 3 CLEAR TO 20, 78
STORE MEMORY VARIABLES
SUM
SUMS VALUES OF SPECIFIED FIELDS INTO MEMORY VARIBLES
DO WHILE COUNTER < 12
    DO CASE
        STORES VALUES OF FIELDS INTO MEMORYVARIABLES CRIME 1 & 2
        STORES RATE, PERCENT CHANGE TO MEMORY VARIABLES
    ENDCASE
    STORE ROUND((CRIME1-CRIME2)/CRIME2)*100),2) TO &PERCNTCHG
    CALCULATES THE PERCENTAGE CHANGE BETWEEN YEARS FOR EACH CRIME
    STORE ROUND (((1000*CRIME1/CURNT_POP),4) TO &CT_RATE
    CALCULATES CRIME RATE PER 1000 INDIVIDUALS
    STORE ROUND (((1000*CRIME2/LAST_POP),4) TO LST_RATE
    CALCULATES CRIME RATE PER 1000 INDIVIDUALS
    IF COUNTER < 8
        TOTALS PRECEEDING AND CURRENT YEAR VALUES
        (INDEX CRIMES ONLY)
        CALCULATES PERCENTAGE CHANGE BETWEEN YEARS
    ENDIF
    STORE COUNTER + 1 TO COUNTER
ENDDO WHILE COUNTER < 12
@ 0,0
SET DEVICE TO PRINT
PRINTS THE FIRST REPORT
STORE T_HOMICIDE + T_RAPE + T_ROBBERY + T_ASSAULT TO C_TOTAL
CALCULATE STATISTICS FOR NEXT REPORT
@ 0,0
PRINT SECOND REPORT
STORE BURGLARY + LARCENY + T_AUTOTHEFT TO PROPCRIME
CALCULATE STATISTICS FOR NEXT REPORT
@ 0,0
PRINT THIRD REPORT
STORE T_NARC + T_SELL + T_DEAL + T_SMOKE TO DRUGTOTAL
@0,0
PRINT FOURTH REPORT
RELEASE ALL
RETURN
```

PROGRAMMER NOTES: This module will format the crime statistics into the four tabular formats required by the user.

Module Description Summary Page

MODULE NAME: Print4.Prg CALLED FROM: Print.Prg	DATA BASES USED: Archives.DBF
THIS MODULE CALLS: none	

```
DO WHILE COUNT < 5
    DO CASE
        (STORES APPROPRIATE HEADING TO MEMORY VARIABLE)
        (STORES APPROPRIATE CRIME TO MEMORY VARIABLE)
    END CASE
    STORE (80-LEN (TRIM (&HEADING))) /2 TO CENTER
    (PRINTS HEADING IN CENTER OF PAGE)
    DO WHILE PLACE < 21
        DO CASE
            (STORES INSTALLATION NAME TO MEMORY VARIABLE)
        END CASE
        LOCATE FOR (STATION = &STATION) .AND. (YEAR = CURNT_YEAR)
        (FINDS RECORD)
        IF POPULATION =0
            CHECKS FOR DIVISION BY ZERO
        ELSE
            STORE 32 TO START
            STORE 30 TO COLUMN
            (COMPUTE CURRENT RATE PER 1000 INDIVIDUALS)
            (ADD RATE TO COLUMN)
            DO WHILE START < DECREMENT
                PRINT GRAPH
            ENDDO WHILE
            (PRINT VALUE OF CURRENT RATE PER 1000 INDIVIDUALS)
        ENDIF POPULATION =0
        NEXT LOCATE SCOPE
        INCREMENT LINE
        INCREMENT PLACE
    ENDDO WHILE PLACE < 21
    IF COUNT >0
        PRINT SCALE 1-10
    ELSE
        PRINT SCALE 10-100
    ENDIF
    PRINT LEGEND
ENDDO WHILE COUNT <5
SET SAFETY ON
EJECT
SET DEVICE TO SCREEN
RELEASE ALL
RETURN
```

Module Description Summary Page

MODULE NAME: Reports.Prg CALLED FROM: Main.Prg	DATA BASES USED: none

THIS MODULE CALLS: Monthly.Prg, Infoannual.Prg, Print.Prg, Audit.Prg

```
STORE 9 TO CHOICE
DO WHILE CHOICE > 5
    CLEAR
    *
    BUILD SCREEN
    *
    DO CASE
        *
        GET USER CHOICE
        *
    OTHERWISE
        *
        DISPLAY ERROR MESSAGE
        STORE 1 TO DELAY
        DO WHILE DELAY < 35
            STORE DELAY + 1 TO DELAY
        ENDDO WHILE DELAY < 35
    ENDCASE
ENDDO WHILE CHOICE > 5
RETURN
```

PROGRAMMER NOTES: This module presents the user with a menu which lists all of the program reports available.

Module Description Summary Page

MODULE NAME: Security.Prg
CALLED FROM: none

DATA BASES USED: Acess.DBF

THIS MODULE CALLS: Main.Prg

```
DO WHILE RIGHT = N
    *
    *
    GET PASSWORD
    IF PASSWORD <> PASSWORD ENTERED
        *
        *
        STORE LOOP =1 + TO LOOP
    ELSE
        STORE Y TO RIGHT
        DO MAIN
    ENDIF
    IF LOOP = 3
        *
        (MESSEGE ACCESS DENIED)
        DO WHILE DELAY < 50
            STORE DELAY + 1 TO DELAY
        ENDDO WHILE DELAY < 50
        CLEAR ALL
        QUIT
    ENDIF
ENDDO WHILE RIGHT = N
RETURN
```

PROGRAMMER NOTES: This program contains the password which allows an authorized user to access the crime statistics database. The user's name as well as access information (files accessed, date & time) is recorded. This information is stored in a file called Access.DBF

Module Description Summary Page (1 of 2)

MODULE NAME: Update.Prg CALLED FROM: Main.Prg	DATA BASES USED: As selected by user (any .DBF File)

THIS MODULE CALLS: GetInfo.Prg

```
DO WHILE REPEAT = Y
    DO WHILE EXIT = N
        DO WHILE CHOICE > 20
            READ
            DO CASE
                *
            OTHERWISE
                CHOICE = 99
            END CASE
        ENDDO WHILE CHOICE > 20
        DO WHILE (MONTH <1) .OR. (MONTH > 12)
            GET DATE
            IF (MONTH < 1) .OR. (MONTH >12)
                ERROR MESSAGE
                DO WHILE DELAY < 50
                STORE 1 + DELAY TO DELAY
                ENDDO WHILE DELAY
            ENDIF
        ENDDO WHILE MONTH
        GET DATE PICTURE, GET VERIFY PICTURE
        IF VERIFY = N
            STORE N TO EXIT
        ENDIF
    ENDDO WHILE EXIT = N
    USE &DBF
    GO TOP
    STORE 0 TO COUNT
```

Module Description Summary Page (2 of 2)

MODULE NAME: Update.Prg CALLED FROM: Main.Prg	DATA BASES USED: As selected by user (any .DBF File)

THIS MODULE CALLS: GetInfo.Prg

```
        DO WHILE COUNT < 28
            STORE COUNT + 1 TO COUNT
            DO CASE
                *
            END CASE
            STORE 00 TO ALL MEMORY VARIABLES
            STORE Y TO VERIFY
            IF VERIFY = Y
                APPEND BLANK
                REPLACE FIELD VALUES WITH MEMORY VARIABLE VALUES
            ELSE
                STORE COUNT -1 TO COUNT
            ENDIF
        ENDDO WHILE COUNT < 28
        GET ANS PICTURE
        IF ANS = Y
            STORE Y TO REPEAT
            CLOSE ALL, CLEAR ALL
        ELSE
            CLOSE ALL,CLEAR ALL
            STORE N TO REPEAT
        ENDIF
    ENDDO WHILE REPEAT = Y
    RETURN
```

PROGRAMMER NOTES: This module allows the user to input monthly crime data which is recieved from the twenty reporting installations. Data is stored in a separate DBF file for each installation.

Module Description Summary Page

MODULE NAME: GetInfo.Prg CALLED FROM: Edit.Prg, Update.Prg	DATA BASES USED: As selected by user (any .DBF File)

THIS MODULE CALLS: None

```
DO WHILE EXIT = N
    DO WHILE CHOICE > 20
        *
        PRESENT MENU OPTIONS
        *
        DO CASE
            SELECT APPROPRIATE DATA BASE
        OTHERWISE
            DISPLAY ERROR MESSAGE
        ENDCASE
    ENDDO WHILE CHOICE > 20
    DO WHILE (MONTH <1 ) OR (MONTH > 12 )
        *
        GET DATE
        IF (MONTH < 1 ) OR (MONTH > 12 )
            PRINT ERROR MESSEGE
        ENDIF
    ENDO WHILE MONTH
    DO WHILE CHOICE > 3
        *
        PRESENT MENU OPTIONS
        DO CASE
            *
            SELECT APPROPRIATE PROGRAM
        OTHERWISE
            DISPLAY ERROR MESSEGE
        ENDCASE
    ENDDO WHILE CHOICE > 3
    *
    *
    VALIDATE INPUT
    *
    IF VERIFY = N
        STORE 99 TO CHOICE
        STORE N TO EXIT
    ELSE
        STORE Y TO EXIT
    ENDIF
ENDO WHILE EXIT = N
RETURN
```

PROGRAMMER NOTES: This module allows the user to specify the station, date of report, and crime statistic which will be updated or edited.

APPENDIX E

```
*********************************************************************************
* Program :      ALCOHOL.PRG                                                    *
* Author :       P. E. PAQUETTE                                                 *
* Date :         10/25/87                                                       *
* Purpose :      THIS MODULE ALLOWS THE USER TO SELECT A                        *
*                SPECIFIC ALCOHOL RELATED OFFENSE FOR UPDATING                  *
*                OR EDITING                                                     *
* Input File :   NONE                                                           *
* Output File :  NONE                                                           *
* Called By :    GETINFO                                                        *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      LOOP, DELAY, CRIME                                             *
*   Global :     NONE                                                           *
*********************************************************************************

SET TALK OFF
SET STATUS OFF
STORE 9 TO LOOP
STORE "                    " TO CRIME
DO WHILE LOOP > 4
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,32 SAY "MARINE CORPS"
@ 4,21 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 6,22 SAY "ALCOHOL RELATED DRIVING OFFENSES"
@ 8,0 SAY CHR(204)
@ 8,79 SAY CHR(185)
@ 8,1 TO 8,78 DOUBLE
@ 10,22 SAY " **** DRIVING WHILE INTOXICATED ****"
@ 12,29 SAY "1.  DWI ON BASE"
@ 13,29 SAY "2.  DWI OFF BASE"
@ 15,20 SAY "**** DRIVING WHILE UNDER THE INFLUENCE ****"
@ 17,29 SAY "3.  DUI ON BASE"
@ 18,29 SAY "4.  DUI OFF BASE"
@ 22,23 SAY "Choose the crime data to be edited"
@ 23,38 GET LOOP PICTURE "9"
READ
  DO CASE
    CASE LOOP = 1
       STORE "DWI ON BASE" TO CRIME
    CASE LOOP = 2
       STORE " DWI OFF BASE" TO CRIME
    CASE LOOP = 3
       STORE "DUI ON BASE" TO CRIME
    CASE LOOP = 4
       STORE "DUI OFF BASE" TO CRIME
  OTHERWISE
    @ 10,17 CLEAR TO 19,55
    @ 10,17 TO 18,55 DOUBLE
    @ 14,25 SAY "ERROR IN INPUT VALUE"
```

```
    @ 15,25 SAY "ENTER A NUMBER LISTED"
    ?? CHR(7)
    STORE 1 TO DELAY
    DO WHILE DELAY <35
      STORE DELAY + 1 TO DELAY
    ENDDO WHILE DELAY <35
  ENDCASE
ENDDO
RETURN
```

```
*************************************************************************************
* Program :      ANNUAL.PRG                                                         *
* Author :       P. E. PAQUETTE                                                     *
* Date :         12/05/86                                                           *
* Purpose :      THIS PROGRAM SEARCHES EACH DATABASE FOR CRIME                      *
*                STATISTICS FOR THE YEAR CHOSEN BY THE USER.                        *
*                THE USER IS PROMPTED FOR THE POPUALTION OF EACH                    *
*                INSTALLATION AS THAT DATABASE IS ACCESSED. ALL                     *
*                THE INFORMATION IS COLLATED AND APPENDED TO THE                    *
*                ARCHIVES DATABASE FILE.                                            *
* Input File :   ALL .DBF FILES                                                     *
* Output File :  ARCHIVES.DBF                                                       *
* Called By :    INFOANNUAL.PRG                                                     *
* Calls :        NONE                                                               *
* Variables                                                                         *
*   Local :      MURDERCNT,MANCOUNT,RAPECOUNT,ARAPECOUNT                            *
*                ARMEDCOUNT,OTHERCOUNT,ASSLTCOUNT,OTWPNCOUNT                        *
*                SIMPLCOUNT,ENTRYCOUNT,TRYCOUNT,DODCOUNT                            *
*                NDODCOUNT,GOVTHEFT,CIVILTHEFT,AUTOCOUNT                            *
*                VANCOUNT,GAUTOCOUNT,NAUTOCOUNT,NARCCOUNT                           *
*                SELLCOUNT,SMOKECOUNT,DEALCOUNT,DWIBASE,DWI                         *
*                DUIBASE,DUI,TINDXTOTAL,CHOICE,USER_POP,USER_ANSR                   *
*                USERREPLY,HOMICIDES,FORCERAPE,ROBBERY,ASSAULT,                     *
*                BURGLARY,LARCENY,AUTOTHEFT,INDEXTOTAL,PROPCRIME,                   *
*                DRUGTOTAL,DUITOTAL,DWITOTAL,DWI,DUI,ALCHLTOTAL,                    *
*                YEAR                                                               *
*   Global :     CURNT_YEAR (passed from infoannual.prg)                            *
*************************************************************************************

@ 10, 3 CLEAR TO 22,78
@ 12, 23 SAY "Compiling Data....."
STORE 0 TO
MURDERCNT,MANCOUNT,RAPECOUNT,ARAPECOUNT,ARMEDCOUNT,OTHERCOUNT
STORE 0 TO
ASSLTCOUNT,OTWPNCOUNT,SIMPLCOUNT,ENTRYCOUNT,TRYCOUNT,DODCOUNT
STORE 0 TO NDODCOUNT,GOVTHEFT,CIVILTHEFT,AUTOCOUNT,VANCOUNT,GAUTOCOUNT
STORE 0 TO NAUTOCOUNT,NARCCOUNT,SELLCOUNT,SMOKECOUNT,DEALCOUNT
STORE 0 TO DWIBASE,DWI,DUIBASE,DUI,TINDXTOTAL
STORE 1 TO CHOICE
STORE 99999 TO USER_POP
STORE "N" TO USER_ANSR
STORE "N" TO USERREPLY
select a
use archive
DO WHILE CHOICE < 21
  DO CASE
    CASE CHOICE =1
        STORE "MCLB ALBANY" TO STATION
        STORE "ALBANY" TO DBF
    CASE CHOICE =2
        STORE "MCLB BARSTOW" TO STATION
        STORE "BARSTOW" TO DBF
    CASE CHOICE =3
        STORE "MCAS BEAUFORT" TO STATION
        STORE "BEAUFORT" TO DBF
```

```
  CASE CHOICE =4
      STORE "CAMP BUTLER" TO STATION
      STORE "BUTLER" TO DBF
  CASE CHOICE =5
      STORE "CHERRY POINT" TO STATION
      STORE "CHERRY" TO DBF
  CASE CHOICE =6
      STORE "MCAS EL TORO" TO STATION
      STORE "ELTORO" TO DBF
  CASE CHOICE =7
      STORE "CAMP ELMORE" TO STATION
      STORE "ELMORE" TO DBF
  CASE CHOICE =8
      STORE "HENDERSON HALL" TO STATION
      STORE "HENDRSON" TO DBF
  CASE CHOICE =9
      STORE "MCAS IWAKUNI" TO STATION
      STORE "IWAKUNI" TO DBF
  CASE CHOICE =10
      STORE "MCAS KANEOHE" TO STATION
      STORE "KANEOHE" TO DBF
  CASE CHOICE =11
      STORE "MCB CAMP LEJEUNE" TO STATION
      STORE "LEJEUNE" TO DBF
  CASE CHOICE =12
      STORE "MCRD PARRIS ISLAND" TO STATION
      STORE "PI" TO DBF
  CASE CHOICE =13
      STORE "MCB CAMP PENDLETON" TO STATION
      STORE "PENDELTN" TO DBF
  CASE CHOICE =14
      STORE "MCDEC QUANTICO" TO STATION
      STORE "QUANTICO" TO DBF
  CASE CHOICE =15
      STORE "MCRD SAN DIEGO" TO STATION
      STORE "SANDIEGO" TO DBF
  CASE CHOICE =16
      STORE "MCAS TUSTIN" TO STATION
      STORE "TUSTIN" TO DBF
  CASE CHOICE =17
      STORE "MCAGCC TWENTYNINE PALMS" TO STATION
      STORE "STUMPS" TO DBF
  CASE CHOICE =18
      STORE "MCAS YUMA" TO STATION
      STORE "YUMA" TO DBF
  CASE CHOICE =19
      STORE "CAMP SMITH" TO STATION
      STORE "SMITH" TO DBF
  CASE CHOICE =20
      STORE "MCAS NEW RIVER" TO STATION
      STORE "NEWRIVER" TO DBF
ENDCASE
DO WHILE USER_ANSR = "N"
  @ 12,23 CLEAR TO 12,70
  @ 10, 23 SAY "PLEASE ENTER THE POPULATION FIGURE"
```

```
   @ 14, 10 SAY "STATION              POULATION          YEAR"
   @ 15, 10 SAY "&STATION"
   @ 15, 33 GET  USER_POP PICTURE "99,999"
   @ 15, 55 SAY CURNT_YEAR PICTURE "9999"
   READ
   @ 20, 18 SAY "HAS THE CORRECT POPULATION COUNT BEEN ENTERED ?"
   @ 21, 38 GET USERREPLY PICTURE "!"
   READ
   IF USERREPLY = "Y"
     USER_ANSR = "Y"
   ELSE
     USER_ANSR = "N"
   ENDIF
ENDDO WHILE USER_ANSR = "N"
SELECT B
USE &DBF
DO WHILE .NOT. EOF()
 if substr(DATE,4,2) = substr(str(CURNT_YEAR),3,2)
 DO CASE
   CASE (CRIME = "MURDER")
        MURDERCNT = F1 + M->MURDERCNT
   CASE (CRIME = "MANSLAUGHTER")
        MANCOUNT = F1 + MANCOUNT
   CASE (CRIME = "RAPE")
        RAPECOUUNT = F1 + RAPECOUUNT
   CASE (CRIME = "ATTEMPTED RAPE")
        ARAPECOUNT = F1 + ARAPECOUNT
   CASE (CRIME = "ROBBERY W/FIRE ARM")
        ARMEDCOUNT = F1 + ARMEDCOUNT
   CASE (CRIME = "ROBBERY W/OTHER WEAPON")
        OTHERCOUNT = F1 + OTHERCOUNT
   CASE (CRIME = "ASSAULT W/FIRE ARM")
        ASSLTCOUNT = F1 + ASSLTCOUNT
   CASE (CRIME = "ASSAULT W/OTHER WEAPON")
        OTWPNCOUNT = F1 + OTWPNCOUNT
   CASE (CRIME = "SIMPLE ASSAULT")
        SIMPLCOUNT = F1 + SIMPLCOUNT
   CASE (CRIME = "ENTRY")
        ENTRYCOUNT = F1 + ENTRYCOUNT
   CASE (CRIME = "ATTEMPTED ENTRY")
        TRYCOUNT = F1 + TRYCOUNT
   CASE (CRIME = "GOVT. PROPERTY")
        DODCOUNT = F1 + DODCOUNT
   CASE (CRIME = "NON-GOVT. PROPERTY")
        NDODCOUNT = F1 + NDODCOUNT
   CASE (CRIME = "LARCENY GOVT. PROPERTY")
        GOVTHEFT = F1 + GOVTHEFT
   CASE (CRIME = "LARCENY NON-GOVT. PROPERTY")
        CIVILTHEFT = F1 + CIVILTHEFT
   CASE (CRIME = "AUTO THEFT")
        AUTOCOUNT = F1 + AUTOCOUNT
   CASE (CRIME = "OTHER VEHICLE THEFT")
        VANCOUNT = F1 + VANCOUNT
   CASE (CRIME = "GOVT. VEHICLE")
        GAUTOCOUNT = F1 + GAUTOCOUNT
```

```
    CASE (CRIME = "NON-GOVT. VEHICLE")
       NAUTOCOUNT = F1 + NAUTOCOUNT
    CASE (CRIME = "NARCOTICS USE & POSSESSION")
       NARCCOUNT = F1 + NARCCOUNT
    CASE (CRIME = "NARCOTICS SALE & TRAFFICKING")
       SELLCOUNT = F1 + SELLCOUNT
    CASE (CRIME = "MARIJUANA USE & POSSESSION")
       SMOKECOUNT + F1 + SMOKECOUNT
    CASE (CRIME = "MAJIJUANA SALE & TRAFFICKING")
       DEALCOUNT = F1 + DEALCOUNT
    CASE (CRIME = "DWI ON BASE")
       DWIBASE = F1 + DWIBASE
    CASE (CRIME = "DWI OFF BASE")
       DWI = F1 + DWI
    CASE (CRIME = "DUI ON BASE")
       DUIBASE = F1 + DUIBASE
    CASE (CRIME = "DUI OFF BASE")
       DUI = F1 + DUI
    ENDCASE
  ENDIF
  SKIP
 ENDDO WHILE NOT EOF
store MURDERCNT + MANCOUNT to HOMICIDES
store RAPECOUNT + ARAPECOUNT to FORCERAPE
store ARMEDCOUNT + OTHERCOUNT to ROBBERY
store ASSLTCOUNT + OTWPNCOUNT + SIMPLCOUNT to ASSAULT
store ENTRYCOUNT + TRYCOUNT + DODCOUNT + NDODCOUNT to BURGLARY
store GOVTHEFT + CIVILTHEFT to LARCENY
store AUTOCOUNT + VANCOUNT + GAUTOCOUNT + NAUTOCOUNT to AUTOTHEFT
store HOMICIDES + FORCERAPE + ROBBERY + ASSAULT + BURGLARY + LARCENY ;
          + AUTOTHEFT to INDEXTOTAL
store homicides + forcerape + robbery + assault to violent
store burglary + larceny + autotheft to propcrime
store narccount  + sellcount + smokecount + dealcount to drugtotal
store duibase + dui to duitotal
store dwibase + dwi to dwitotal
store duitotal + dwitotal to alcholtotl
if VAL(substr("&DATE",4,2)) > 80
  store "19" + substr("&DATE",4,2) to year
else
  store "20" + substr("&DATE",4,2) to year
endif
select a
append blank
replace station with "&station"
replace homicides with homicides
replace FORCERAPE with FORCERAPE
REPLACE ROBBERY WITH ROBBERY
replace assault with assault
replace burglary with burglary
replace larceny with larceny
replace autotheft with autotheft
replace narccount with narccount
replace sellcount with sellcount
replace smokecount with smokecount
```

```
replace dealcount with dealcount
replace dwibase with dwibase
replace dwi with dwi
replace duibase with duibase
replace dui with dui
replace indextotal with indextotal
REPLACE VIOLENT WITH VIOLENT
REPLACE PROPCRIME WITH PROPCRIME
replace drugtotal with drugtotal
replace totaldui with DUITOTAL
replace totaldwi with DWITOTAL
replace BOOZETOTAL with ALCHOLTOTL
replace year with year
replace population with USER_POP
STORE 20 TO COUNT
STORE COUNT - CHOICE TO COUNT
@ 10, 3 CLEAR TO 23, 78
@ 12, 23 SAY "&STATION is done, only" + STR(COUNT,4) + " to go."
USER_INDX = CHOICE + 1
release all except USER*
CHOICE = USER_INDX
USER_ANSR = "N"
ENDDO WHILE CHOICE < 21
CLOSE DATABASES
RELEASE ALL EXCEPT USER*
RETURN
```

```
*********************************************************************************
* Program :      AUDIT.PRG                                                      *
* Author :       P. E. PAQUETTE                                                 *
* Date :         2/15/87                                                        *
* Purpose :      THIS MODULE PROVIDES THE DATABASE ADMINISTRATOR                 *
*                WITH AN AUDIT TRAIL OF ACCESSES AND MODIFICATIONS              *
*                TO THE SYSTEM                                                  *
* Input File  :  ACCESS.DBF                                                     *
* Output File :  UPDATED ACCESS.DBF                                             *
* Called By :    REPORTS.PRG                                                    *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      ANS                                                            *
*   Global :     NONE                                                           *
*********************************************************************************

STORE 0 TO DELAY
STORE "N" TO ANS
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3, 34 SAY "MARINE CORPS"
@ 4, 23 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 6, 33 SAY "ACCESS SUMMARY"
@ 8, 0  SAY CHR(204)
@ 8, 1 TO 8, 78 DOUBLE
@ 8, 79 SAY CHR(185)
USE ACCESS
@ 10, 27 SAY "DATE"
@ 10, 43 SAY "TIME"
@ 11, 5  SAY "USER"
@ 11, 28 SAY "OF"
@ 11, 44 SAY "OF"
@ 11, 62 SAY "DATABASE"
@ 12, 5  SAY "NAME"
@ 12, 26 SAY "ACCESS"
@ 12, 42 say "ACCESS"
@ 12, 61 SAY "OPERATIONS"
LINE = 14
DO WHILE .NOT. EOF()
  @ LINE, 5  SAY LOGUSER
  @ LINE, 25 SAY LOGDATE
  @ LINE, 41 SAY LOGTIME
  @ LINE, 63 SAY LOGCHOICEC
  SKIP
  LINE = LINE + 1
  IF LINE = 22
    LINE = 14
    @ 23,28 SAY "PRESS ANY KEY TO CONTINUE"
    READ
  ENDIF
ENDDO WHILE NOT EOF
@ 23,28 CLEAR TO 23, 78
DO WHILE DELAY < 100
     DELAY = DELAY + 1
ENDDO WHILE DELAY
```

```
@ 14, 1 CLEAR TO 22,78
@ 15, 20 SAY "DO YOU DESIRE A PRINTOUT OF THIS REPORT?"
@ 17,39 GET ANS PICTURE "!"
READ
IF ANS = "Y"
  @ 10, 17 CLEAR TO 19,55
  @ 10, 17 TO 18, 55 DOUBLE
  @ 14, 28 SAY "PERPARE PRINTER"
  READ
  SET DEVICE TO PRINT
  @ 0,0
  @ 3, 34 SAY "MARINE CORPS"
  @ 4, 23 SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 6, 33 SAY "ACCESS SUMMARY"
  @ 10, 27 SAY "DATE"
  @ 10, 43 SAY "TIME"
  @ 11, 5  SAY "USER"
  @ 11, 28 SAY "OF"
  @ 11, 44 SAY "OF"
  @ 11, 62 SAY "DATABASE"
  @ 12, 5  SAY "NAME"
  @ 12, 26 SAY "ACCESS"
  @ 12, 42 say "ACCESS"
  @ 12, 61 SAY "OPERATIONS"
  LINE = 14
  GO TOP
  DO WHILE .NOT. EOF()
    IF LINE > 55
      EJECT
      LINE = 5
    ENDIF
    @ LINE, 5  SAY LOGUSER
    @ LINE, 25 SAY LOGDATE
    @ LINE, 41 SAY LOGTIME
    @ LINE, 63 SAY LOGCHOICEC
    SKIP
    LINE = LINE + 1
  ENDDO WHILE NOT EOF
  EJECT
  SET DEVICE TO SCREEN
ENDIF ANS = Y
@ 23, 28 SAY "RETURNING TO MAIN MENU    "
CLOSE ALL DATABASES
RELEASE ALL EXCEPT USER*
RETURN
```

```
*********************************************************************************
* Program :      DELETE.PRG                                                     *
* Author :       P. E. PAQUETTE                                                 *
* Date :         11/11/87                                                       *
* Purpose :      This program allows the user to delete redundant               *
*                or erroneous records from the database                         *
* Input File :   as specified by user                                           *
* Output File :  updated .DBF file                                              *
* Called By :    MAON.PRG                                                       *
* Calls :        GETINFO.PRG                                                    *
* Variables                                                                     *
*   Local :      EXIT, DELAY,                                                   *
*   Global :     NONE                                                           *
*********************************************************************************

STORE "N" TO EXIT
DO WHILE EXIT = "N"
DO GETINFO
STORE 9 TO CHOICE
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,33 SAY "MARINE CORPS"
@ 4,22 SAY "CRIME STATISTICS REPORTING PROGRAM"
SET COLOR TO GR+/R*
@ 6,22 SAY "      SEARCHING DATABASE "
SET COLOR TO GR+/R,W/R,N
@ 8,0 SAY CHR(204)
@ 8,79 SAY CHR(185)
@ 8,1 TO 8,78 DOUBLE
@ 10, 26 SAY "INSTALLATION NAME:  &STATION"
@ 12, 26 SAY "DATE OF REPORT:  &DATE"
USE &DBF
GO TOP
LOCATE FOR (CRIME = "&CRIME") .AND. (DATE = "&DATE")
     IF .NOT. FOUND() .AND. EOF()
        @ 14, 18 CLEAR TO 18,61
        @ 14, 18 TO 18,61   DOUBLE
        @ 18, 29 SAY "*** NO MATCH FOUND***"
        @ 15,20 SAY "ENSURE A VALID DATE & CRIME WERE ENTERED"
        ?? CHR(7)
        STORE 1 TO DELAY
        DO WHILE DELAY < 50
         STORE DELAY + 1 TO DELAY
        ENDDO WHILE DELAY < 50
        CLOSE DATABASES
        RELEASE ALL EXCEPT USER*
     ELSE
        CLEAR
        @ 1, 1 TO 23, 79 DOUBLE
        @ 5, 1 SAY CHR(199)
        @ 5, 79 SAY CHR(182)
        @ 5, 2 TO 5, 78
        @ 2, 26 SAY "INSTALLATION:  &STATION"
        @ 3, 26 SAY "DATE OF REPORT:  &DATE"
        @ 4, 20 SAY "CLASSIFICATION OF OFFENSE:  &CRIME"
```

```
        SET COLOR TO GR+/R*
        @ 10, 25 SAY "*** READY FOR DELETION ***"
        SET COLOR TO GR+/R,W/R,N
        @ 12, 25 SAY "1.  DELETE MONTYLY REPORT"
        @ 14, 25 SAY "2.  DELETE OFFENSE RECORD ONLY"
        @ 16, 25 SAY   "PRESS ANY KEY TO ABORT DELETION"
        @ 18, 37 GET CHOICE PICTURE "9"
        READ
        DO CASE
          CASE CHOICE = 1
            CLEAR
            SET COLOR TO GR+/R*
            @ 12, 29 SAY "DELETION IN PROGRESS"
            GO TOP
            DELETE ALL FOR DATE = "&DATE"
            PACK
            CLOSE DATABASES
            RELEASE ALL EXCEPT USER*
            SET COLOR TO GR+/R,W/R,N
            STORE "Y" TO EXIT
          CASE CHOICE = 2
            CLEAR
            SET COLOR TO GR+/R*
            @ 12, 29 SAY "DELETION IN PROGRESS"
            DELETE ALL FOR (CRIME = "&CRIME") .AND. (DATE="&DATE")
            PACK
            CLOSE DATABASES
            CLEAR ALL except user*
            SET COLOR TO GR+/R,W/R,N
            STORE "Y" TO EXIT
          OTHERWISE
            RELEASE ALL EXCEPT USER*
            CLOSE DATABASES
            STORE "Y" TO EXIT
          ENDCASE
      ENDIF NOT FOUND
ENDDO WHILE EXIT = "N"
RETURN
```

```
*********************************************************************************
* Program :      DRUGS.PRG                                                      *
* Author :       P. E. PAQUETTE                                                 *
* Date..:        10/25/87                                                       *
* Purpose :      THIS MODULE ALLOWS THE USER TO SELECT A                        *
*                SPECIFIC DRUG RELATED OFFENSE FOR UPDATING OR                  *
*                EDITING                                                        *
* Input File :   NONE                                                           *
* Output File :  NONE                                                           *
* Called By..:   GETINFO                                                        *
* Calls..:       NONE                                                           *
* Variables                                                                     *
*   Local..:     LOOP, DELAY, CRIME                                             *
*   Global..:    NONE                                                           *
*********************************************************************************

STORE 9 TO LOOP
STORE "                    " TO CRIME
DO WHILE LOOP > 4
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,31 SAY "MARINE CORPS"
@ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 6,27 SAY "DRUG RELATED OFFENSES"
@ 8,0 SAY CHR(204)
@ 8,79 SAY CHR(185)
@ 8,1 TO 8,78 DOUBLE
@  9, 24  SAY "**** NARCOTICS/DRUGS ****"
@ 11, 23  SAY "1.  Use & Possession"
@ 12, 23  SAY "2.  Sale & Trafficking"
@ 15, 24  SAY "****   MARIJUANA    ****"
@ 17, 23  SAY "3.  Use & Possession"
@ 18, 23  SAY "4.  Sale & Trafficking"
@ 21, 14  SAY "ENTER THE NUMBER OF THE CRIME DATA TO BE EDITED"
@ 23, 35  GET LOOP PICTURE "9"
READ
  DO CASE
    CASE LOOP = 1
       STORE "NARCOTICS USE & POSSESSION" TO CRIME
    CASE LOOP = 2
       STORE " NARCOTICS SALE & TRAFFICKING" TO CRIME
    CASE LOOP = 3
       STORE "MARIJUANA USE & POSSESSION" TO CRIME
    CASE LOOP = 4
       STORE "MARIJUANA SALE & TRAFFICKING" TO CRIME
  OTHERWISE
    @ 10,17 CLEAR TO 19,55
    @ 10,17 TO 18,55 DOUBLE
    @ 14,25 SAY "ERROR IN INPUT VALUE"
    @ 15,25 SAY "ENTER A NUMBER LISTED"
    ?? CHR(7)
    STORE 1 TO DELAY
    DO WHILE DELAY <35
      STORE DELAY + 1 TO DELAY
    ENDDO WHILE DELAY <35
```

```
  ENDCASE
ENDDO
RETURN
```

```
*********************************************************************************
* Program :      EDIT.PRG                                                       *
* Author :       P. E. PAQUETTE                                                 *
* Date :         11/11/87                                                       *
* Purpose :      This program allows the user to select a                       *
*                specific record for editing                                    *
* Input File :   DBF specified by user                                          *
* Output File :  updated .DBF file                                              *
* Called By :    MAIN.PRG                                                       *
* Calls :        GETINFO.PRG                                                    *
* Variables                                                                     *
*   Local :      REPEAT, EXIT, DELAY, VERIFY, TF-1 through TF-28                *
*                TDATE, TCRIME                                                  *
*   Global :     NONE                                                           *
*********************************************************************************

STORE "Y" TO REPEAT
STORE "N" TO EXIT
DO WHILE EXIT = "N"
DO GETINFO
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,33 SAY "MARINE CORPS"
@ 4,22 SAY "CRIME STATISTICS REPORTING PROGRAM"
SET COLOR TO GR+/R*
@ 6,22 SAY "      SEARCHING DATABASE "
SET COLOR TO GR+/R,W/R,N
@ 8,0 SAY CHR(204)
@ 8,79 SAY CHR(185)
@ 8,1 TO 8,78 DOUBLE
@ 10, 26 SAY "INSTALLATION NAME:  &STATION"
@ 12, 26 SAY "DATE OF REPORT:  &DATE"
USE &DBF
SET EXACT ON
LOCATE FOR (CRIME = "&CRIME") .AND. (DATE = "&DATE")
     IF .NOT. FOUND() .AND. EOF()
       @ 14, 18 CLEAR TO 18,55
       @ 14, 18 TO 18,61   DOUBLE
       @ 18, 29 SAY "*** NO MATCH FOUND***"
       @ 15,20 SAY "ENSURE A VALID DATE & CRIME WERE ENTERED"
       ?? CHR(7)
       STORE 1 TO DELAY
       DO WHILE DELAY < 50
        STORE DELAY + 1 TO DELAY
       ENDDO WHILE DELAY < 50
       CLOSE DATABASES
       RELEASE ALL EXCEPT USER*
     ELSE
       DO WHILE REPEAT = "Y"
        STORE F1 TO TF1
        STORE F2 TO TF2
        STORE F3 TO TF3
        STORE F4 TO TF4
        STORE F5 TO TF5
        STORE F6 TO TF6
```

```
STORE F9 TO TF9
STORE F10 TO TF10
STORE F11 TO TF11
STORE F12 TO TF12
STORE F13 TO TF13
STORE F14 TO TF14
STORE F15 TO TF15
STORE F16 TO TF16
STORE F17 TO TF17
STORE F18 TO TF18
STORE F19 TO TF19
STORE F20 TO TF20
STORE F21 TO TF21
STORE F22 TO TF22
STORE F23 TO TF23
STORE F24 TO TF24
STORE F25 TO TF25
STORE F26 TO TF26
STORE F27 TO TF27
STORE F28 TO TF28
STORE DATE TO TDATE
STORE CRIME TO TCRIME
CLEAR
@  1,  1  TO 23, 79    DOUBLE
@  5,  1  SAY CHR(199)
@  5, 79  SAY CHR(182)
@  5,  2  TO  5, 78
@  2, 26  SAY "INSTALLATION:  &STATION"
@  3, 26  SAY "DATE OF REPORT:  &DATE"
@  4, 20  SAY "CLASSIFICATION OF OFFENSE:  &CRIME"
@  6,  2  SAY "NUMBER OF OFFENSES ON BASE:"
@  6, 30  GET  TF1 PICTURE '99'
@  6, 37  SAY "OFF BASE:"
@  6, 47  GET  TF2 PICTURE '99'
@  7,  2  SAY "NUMBER OF UNFOUNDED FALSE REPORTS ON BASE:"
@  7, 45  GET  TF3 PICTURE '99'
@  7, 50  SAY "OFF BASE:"
@  7, 60  GET  TF4 PICTURE '99'
@  8, 2  SAY "NUMBER CLEARED BY ARREST:"
@  8, 28  GET  TF5 PICTURE '99'
@  9,  2  SAY "NUMBER INVESTIGATED BY NIS:"
@  9, 30  GET  TF6 PICTURE '99'
@ 11, 41  SAY "PERPETRATOR       VICTIM"
@ 12,  2  SAY "USMC:"
@ 13,  2  SAY "OTHER SERVICES:"
@ 14,  2  SAY "DEPENDANTS:"
@ 15,  2  SAY "DOD CIVILIAN PERSONNEL:"
@ 16,  2  SAY "MALES:"
@ 17,  2  SAY "FEMALES:"
@ 18,  2  SAY "CAUCASIAN:"
@ 19,  2  SAY "NEGRO:"
@ 20,  2  SAY "ALL OTHERS:"
@ 12, 45 GET TF9   PICTURE '99'
@ 13, 45 GET TF10  PICTURE '99'
@ 14, 45 GET TF11  PICTURE '99'
```

```
@ 15, 45 GET TF12  PICTURE '99'
@ 16, 45 GET TF13  PICTURE '99'
@ 17, 45 GET TF14  PICTURE '99'
@ 18, 45 GET TF15  PICTURE '99'
@ 19, 45 GET TF16  PICTURE '99'
@ 20, 45 GET TF17  PICTURE '99'
@ 12, 61 GET TF18  PICTURE '99'
@ 13, 61 GET TF19  PICTURE '99'
@ 14, 61 GET TF20  PICTURE '99'
@ 15, 61 GET TF21  PICTURE '99'
@ 16, 61 GET TF22  PICTURE '99'
@ 17, 61 GET TF23  PICTURE '99'
@ 18, 61 GET TF24  PICTURE '99'
@ 19, 61 GET TF25  PICTURE '99'
@ 20, 61 GET TF26  PICTURE '99'
@ 21,  2  SAY "DRUG INVOLVEMENT:"
@ 21, 45  GET  TF27 PICTURE '99'
@ 22,  2  SAY "ALCOHOL INVOLVEMENT:"
@ 22, 45  GET  TF28 PICTURE '99'
READ
STORE "Y" TO VERIFY
@ 23,24  CLEAR TO 23,51
@ 23, 24 SAY "ARE ALL ENTRIES CORRECT ?"
@ 23, 50 GET VERIFY PICTURE "Y"
READ
IF VERIFY = "Y"
   STORE "Y" TO EXIT
   STORE "N" TO REPEAT
   REPLACE F1 WITH TF1
   REPLACE F2 WITH TF2
   REPLACE F3 WITH TF3
   REPLACE F4 WITH TF4
   REPLACE F5 WITH TF5
   REPLACE F6 WITH TF6
   REPLACE F9 WITH TF9
   REPLACE F10 WITH TF10
   REPLACE F11 WITH TF11
   REPLACE F12 WITH TF12
   REPLACE F13 WITH TF13
   REPLACE F14 WITH TF14
   REPLACE F15 WITH TF15
   REPLACE F16 WITH TF16
   REPLACE F17 WITH TF17
   REPLACE F18 WITH TF18
   REPLACE F19 WITH TF19
   REPLACE F20 WITH TF20
   REPLACE F21 WITH TF21
   REPLACE F22 WITH TF22
   REPLACE F23 WITH TF23
   REPLACE F24 WITH TF24
   REPLACE F25 WITH TF25
   REPLACE F26 WITH TF26
   REPLACE F27 WITH TF27
   REPLACE F28 WITH TF28
  REPLACE DATE WITH TDATE
```

```
            REPLACE CRIME WITH TCRIME
          ELSE
            STORE "N" TO EXIT
            STORE "Y" TO REPEAT
          ENDIF
        ENDDO WHILE REPEAT ="Y"
      ENDIF NOT FOUND
ENDDO WHILE EXIT= "N"
RELEASE ALL EXCEPT USER*
CLOSE DATABASES
RETURN
```

```
*******************************************************************************
* Program :      GETINFO.PRG                                                  *
* Author :       PAQUETTE                                                     *
* Date :         10/10/87                                                     *
* Purpose :      THIS MODULE ALLOWS THE USER TO LOCATE A                      *
*                SPECIFIC RECORD FOR UPDATE OR EDITING                        *
* Input File :   DETERMINED BY THE VALUE OF USER INPUT                        *
* Output File :  DETERMINED BY THE VALUE OF USER INPUT                        *
* Called By :    UPDATE.PRG, DELETE.PRG                                       *
* Calls :        INDXCRMS.PRG, ALCOHOL.PRG, DRUGS.PRG                        *
* Variables                                                                   *
*   Local :      EXIT, CHOICE, STATION, DELAY, DBF, MONTH,                    *
*                DATE, VERIFY                                                 *
*   Global :     DATE, STATION, CRIME, DBF                                    *
*******************************************************************************

PUBLIC DATE, STATION, CRIME, DBF
STORE 99 TO CHOICE
STORE "              " TO STATION
STORE "       " TO DBF
STORE "N" TO EXIT
STORE 23 TO MONTH
DO WHILE EXIT = "N"
  DO WHILE choice > 20
  CLEAR
  @ 0,0 TO 24,79 DOUBLE
  @ 3,34 SAY "MARINE CORPS"
  @ 4,23 SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 6,30 SAY "FILE SELECTION MENU"
  @ 8,0 SAY CHR(204)
  @ 8,79 SAY CHR(185)
  @ 8,1 TO 8,78 DOUBLE
  @ 10,15 SAY "1. MCLB ALBANY          11. MCB CAMP LEJEUNE"
  @ 11,15 SAY "2. MCLB BARSTOW          12. MCRD PARRIS ISLAND"
  @ 12,15 SAY "3. MCAS BEAUFORT         13. MCB CAMP PENDLETON"
  @ 13,15 SAY "4. CAMP BUTLER          14. MCDEC QUANTICO"
  @ 14,15 SAY "5. MCAS CHERRY POINT      15. MCRD SAN DIEGO"
  @ 15,15 SAY "6. MCAS EL TORO          16. MCAS TUSTIN"
  @ 16,15 SAY "7. CAMP ELMORE           17. MCAGCC 29 PALMS"
  @ 17,15 SAY "8. HENDERSON HALL         18. MCAS YUMA"
  @ 18,15 SAY "9. MCAS IWAKUNI          19. CAMP SMITH"
  @ 19,14 SAY "10. MCAS KANEOHE          20. MCAS NEW RIVER"
  @ 21,20 SAY "SELECT THE STATION YOU WISH TO WORK WITH"
  @ 22,20 SAY "   ENTER '99' TO RETURN TO MAIN MENU"
  @ 23,37 GET CHOICE PICTURE "99"
  READ
  DO CASE
    CASE CHOICE =1
        STORE "MCLB ALBANY" TO STATION
        STORE "ALBANY" TO DBF
    CASE CHOICE =2
        STORE "MCLB BARSTOW" TO STATION
        STORE "BARSTOW" TO DBF
    CASE CHOICE =3
        STORE "MCAS BEAUFORT" TO STATION
```

```
    STORE "BEAUFORT" TO DBF
CASE CHOICE =4
    STORE "CAMP BUTLER" TO STATION
    STORE "BUTLER" TO DBF
CASE CHOICE =5
    STORE "CHERRY POINT" TO STATION
    STORE "CHERRY" TO DBF
CASE CHOICE =6
    STORE "MCAS EL TORO" TO STATION
    STORE "ELTORO" TO DBF
CASE CHOICE =7
    STORE "CAMP ELMORE" TO STATION
    STORE "ELMORE" TO DBF
CASE CHOICE =8
    STORE "HENDERSON HALL" TO STATION
    STORE "HENDRSON" TO DBF
CASE CHOICE =9
    STORE "MCAS IWAKUNI" TO STATION
    STORE "IWAKUNI" TO DBF
CASE CHOICE =10
    STORE "MCAS KANEOHE" TO STATION
    STORE "KANEOHE" TO DBF
CASE CHOICE =11
    STORE "MCB CAMP LEJEUNE" TO STATION
    STORE "LEJEUNE" TO DBF
CASE CHOICE =12
    STORE "MCRD PARRIS ISLAND" TO STATION
    STORE "PI" TO DBF
CASE CHOICE =13
    STORE "MCB CAMP PENDLETON" TO STATION
    STORE "PENDELTN" TO DBF
CASE CHOICE =14
    STORE "MCDEC QUANTICO" TO STATION
    STORE "QUANTICO" TO DBF
CASE CHOICE =15
    STORE "MCRD SAN DIEGO" TO STATION
    STORE "SANDIEGO" TO DBF
CASE CHOICE =16
    STORE "MCAS TUSTIN" TO STATION
    STORE "TUSTIN" TO DBF
CASE CHOICE =17
    STORE "MCAGCC TWENTYNINE PALMS" TO STATION
    STORE "STUMPS" TO DBF
CASE CHOICE =18
    STORE "MCAS YUMA" TO STATION
    STORE "YUMA" TO DBF
CASE CHOICE =19
    STORE "CAMP SMITH" TO STATION
    STORE "SMITH" TO DBF
CASE CHOICE =20
    STORE "MCAS NEW RIVER" TO STATION
    STORE "NEWRIVER" TO DBF
CASE CHOICE = 99
    RETURN
OTHERWISE
```

```
        @ 10,17 CLEAR TO 19,55
        @ 10,17 TO 18,55 DOUBLE
        @ 14,25 SAY "ERROR IN INPUT VALUE"
        @ 15,25 SAY "ENTER A NUMBER LISTED"
        ?? CHR(7)
        STORE 1 TO DELAY
        DO WHILE DELAY < 50
          STORE DELAY +1 TO DELAY
        ENDDO WHILE DELAY < 50
        CHOICE = 99
    ENDCASE
  ENDDO WHILE CHOICE > 20
  STORE "MM/YY" TO DATE
  STORE 23 TO MONTH
  DO WHILE (MONTH < 1) .OR. (MONTH > 12)
    @ 9,01 CLEAR TO 23,78
    @ 14,25 SAY "WHAT IS THE DATE OF THE REPORT"
    @ 15,25 SAY " THAT YOU WISH TO WORK WITH?"
    @ 16,38 GET DATE PICTURE "99/99"
    READ
    MONTH = VAL(SUBSTR(DATE,1,2))
    IF (MONTH < 1) .OR. (MONTH > 12)
      @ 9,1 CLEAR TO 23,78
      @ 12, 18 SAY "  SORRY ONLY TWELVE MONTHS IN A YEAR !"
      @ 14, 18 SAY "PICK A NUMBER BETWEEN 1 AND 12 INCLUSIVE"
      ?? CHR(7)
      STORE 1 TO DELAY
      DO WHILE DELAY < 50
        STORE DELAY +1 TO DELAY
      ENDDO WHILE DELAY < 50
    ENDIF
  ENDDO WHILE MONTH
  CHOICE = 9
  DO WHILE CHOICE > 3
    CLEAR
    @ 0,0 TO 24,79 DOUBLE
    @ 3,31 SAY "MARINE CORPS"
    @ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
    @ 6,20 SAY "    CLASSIFICATION OF OFFENSE "
    @ 8,0 SAY CHR(204)
    @ 8,1 TO 8,78 DOUBLE
    @ 8,79 SAY CHR(185)
    @ 10,19 SAY "1.  INDEX CRIMES "
    @ 11,19 SAY "      A.  CRIMES AGAINST PERSONS"
    @ 12,19 SAY "      B.  CRIMES AGAINST PROPERTY"
    @ 14,19 SAY "2.  DRUG RELATED OFFENSES"
    @ 15,19 SAY "      A.  NARCOTICS/DANGEROUS DRUGS"
    @ 16,19 SAY "      B.  MARIJUANA"
    @ 18,19 SAY "3.  ALCOHOL RELATED OFFENSES"
    @ 19,19 SAY "      A.  DUI"
    @ 20,19 SAY "      B.  DWI"
    @ 22,8 SAY "CHOSE THE NUMBER OF THE REPORTING CATEGORY FOR THE CRIME
DATA"
    SET COLOR TO R/R,W/R,N
    @ 23,35 GET CHOICE PICTURE "9"
```

```
    READ
    SET COLOR TO GR+/R,W/R,N
    DO CASE
      CASE CHOICE = 1
          DO INDXCRMS
      CASE CHOICE = 2
          DO DRUGS
      CASE CHOICE = 3
          DO ALCOHOL
    OTHERWISE
      @ 10,17 CLEAR TO 19,55
      @ 10,17 TO 18,55 DOUBLE
      @ 14,25 SAY "ERROR IN INPUT VALUE"
      @ 15,25 SAY "ENTER A NUMBER LISTED"
      ?? CHR(7)
      STORE 1 TO DELAY
      DO WHILE DELAY < 50
        STORE DELAY +1 TO DELAY
      ENDDO WHILE DELAY < 50
      CHOICE = 9
    ENDCASE
  ENDDO WHILE CHOICE > 3
  CLEAR
  @ 0,0 TO 24,79 DOUBLE
  @ 3,31 SAY "MARINE CORPS"
  @ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 6,28 SAY "VALIDATION OF INPUT"
  @ 8,0 SAY CHR(204)
  @ 8,1 TO 8,78 DOUBLE
  @ 8,79 SAY CHR(185)
  @ 10,7 SAY "THE INFORMATION WHICH YOU HAVE PROVIDED WILL BE USED TO
SEARCH"
  @ 11,7 SAY "THE CRIME REPORTING STATISTICS DATA BASE.  PLEASE CHECK THE"
  @ 12,7 SAY "INFORMATION BELOW TO ENSURE THAT YOUR ENTRIES ARE CORRECT"
  @ 16,26 SAY "INSTALLATION NAME:  &STATION"
  @ 18,26 SAY "DATE OF REPORT: &DATE "
  @ 20,26 SAY "SPECIFIC OFFENSE: &CRIME"
  @ 22,26 SAY "ARE ALL ENTRIES CORRECT?"
  STORE "Y" TO VERIFY
  @ 23,37 GET VERIFY PICTURE "!"
  READ
  IF VERIFY = "Y"
    STORE "Y" TO EXIT
  ELSE
    STORE "N" TO EXIT
    STORE 99 TO CHOICE
  ENDIF
ENDDO WHILE EXIT = N
RETURN
```

```
*********************************************************************************
* Program :      INDXCRMS.PRG                                                   *
* Author :       P. E. PAQUETTE                                                 *
* Date :         10/25/87                                                       *
* Purpose :      THIS MODULE PRESENTS THE USER WITH A LIST OF                   *
*                ALL INDEX CRIMES.  IT STORES THE USER'S CHOICE                 *
*                AS THE CRIME STATISTIC WHICH WILL BE UPDATED                   *
* Input File :   NONE                                                           *
* Output File :  NONE                                                           *
* Called By :    GETINFO                                                        *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      LOOP                                                           *
*   Global :     NONE                                                           *
*********************************************************************************

STORE 99 TO LOOP
STORE "                    " TO CRIME
DO WHILE LOOP > 19
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,31 SAY "MARINE CORPS"
@ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 6,31 SAY "INDEX CRIMES"
@ 7,0 SAY CHR(204)
@ 7,79 SAY CHR(185)
@ 7,1 TO 7,78 DOUBLE
@  8,  2  SAY "CRIMES AGAINST PERSONS                CRIMES AGTAINST PROPERTY"
@ 10,  3  SAY "1.  Murder                        10.  Entry"
@ 11,  3  SAY "2.  Manslaughter                  11.  Attempted Entry"
@ 12,  3  SAY "3.  Rape                          12.  Govt. Property"
@ 13,  3  SAY "4.  Attempted Rape                13.  Non-Govt. Property"
@ 14,  3  SAY "5.  Robbery w/Fire Arm            14.  Larceny of DOD Property"
@ 15,  3  SAY "6.  Robbery w/Other Weapon        15.  Larceny of Non-DOD Property"
@ 16,  3  SAY "7.  Assault W/Fire Arm            16.  Auto Theft"
@ 17,  3  SAY "8.  Assault w/Other Weapon        17.  Other Vehicle Theft"
@ 18,  3  SAY "9.  Simple Assault                18.  Govt. Vehicle Theft"
@ 19,  3  SAY "                                 19.  Non-Govt. Vehicle Theft"
@ 21, 16  SAY "CHOSE THE NUMBER OF THE CRIME DATA TO BE EDITED"
SET COLOR TO R/R,W/R,N
@ 23, 35  GET LOOP PICTURE "99"
READ
SET COLOR TO GR+/R,W/R,N
  DO CASE
    CASE LOOP = 1
       STORE "MURDER" TO CRIME
    CASE LOOP = 2
       STORE "MANSLAUGHTER" TO CRIME
    CASE LOOP = 3
       STORE "RAPE" TO CRIME
    CASE LOOP = 4
       STORE "ATTEMPTED RAPE" TO CRIME
    CASE LOOP = 5
       STORE "ROBBERY W/FIRE ARM" TO CRIME
    CASE LOOP = 6
```

```
        STORE "ROBBERY W/OTHER WEAPON" TO CRIME
    CASE LOOP = 7
        STORE "ASSAULT W/FIRE ARM" TO CRIME
    CASE LOOP = 8
        STORE "ASSAULT W/OTHER WEAPON" TO CRIME
    CASE LOOP = 9
        STORE "SIMPLE ASSAULT" TO CRIME
    CASE LOOP = 10
        STORE "ENTRY" TO CRIME
    CASE LOOP = 11
        STORE "ATTEMPTED ENTRY" TO CRIME
    CASE LOOP = 12
        STORE "GOVT. PROPERTY" TO CRIME
    CASE LOOP = 13
        STORE "NON-GOVT. PROPERTY" TO CRIME
    CASE LOOP = 14
         STORE "LARCENY OF DOD PROPERTY" TO CRIME
    CASE LOOP = 15
        STORE  "LARCENY OF NON-DOD PROPERTY" TO CRIME
    CASE LOOP = 16
        STORE "AUTO THEFT" TO CRIME
    CASE LOOP = 17
        STORE "OTHER VEHICLE THEFT" TO CRIME
    CASE LOOP = 18
        STORE "GOVT. VEHICLE" TO CRIME
    CASE LOOP = 19
        STORE "NON-GOVT. VEHICLE" TO CRIME
  OTHERWISE
    @ 10,17 CLEAR TO 19,55
    @ 10,17 TO 18,55 DOUBLE
    @ 14,25 SAY "ERROR IN INPUT VALUE"
    @ 15,25 SAY "ENTER A NUMBER LISTED"
    ?? CHR(7)
    STORE 1 TO DELAY
    DO WHILE DELAY <35
      STORE DELAY + 1 TO DELAY
    ENDDO WHILE DELAY <35
  ENDCASE
 ENDDO WHILE LOOP > 19
RETURN
```

```
*********************************************************************************
* Program :      INFOANNUAL.PRG                                                 *
* Author :       P. E. PAQUETTE                                                 *
* Date :         12/16/86                                                       *
* Purpose :      THIS PROGRAM GETS THE YEAR OF THE ANNUAL REPORT                *
*                AND CHECKS TO SEE IF A REPORT ALREADY EXISTS                   *
*                FOR THAT YEAR.  IF A REPORT EXISTS, THE USER                   *
*                IS GIVEN THE OPTION OF OVER-WRITING THE REPORT                 *
*                OR EXITING TO THE REPORTS MENU                                 *
* Input File :   ARCHIVES.DBF                                                   *
* Output File :  NONE                                                           *
* Called By :    REPORTS.PRG                                                    *
* Calls :        ANNUAL.PRG                                                     *
* Variables                                                                     *
*   Local :      CONTINUE, REPEAT, CONFIRM, CURNT_YEAR                          *
*   Global :                                                                    *
*********************************************************************************

STORE "N" TO CONTINUE, REPEAT, CONFIRM
STORE 1999 TO CURNT_YEAR
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,33 SAY "MARINE CORPS"
@ 4,22 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 6,22 SAY "      REPORTS GENERATION       "
@ 8,0 SAY CHR(204)
@ 8,1 TO 8,78 DOUBLE
@ 8,79 SAY CHR(185)
DO WHILE REPEAT <> "Y"
  @ 10, 3 CLEAR TO 20,78
  @ 12, 21 SAY "What is the year of this annual report?"
  @ 13, 36 GET CURNT_YEAR PICTURE "9999"
  READ
  @ 10, 3 CLEAR TO 20, 78
  @ 12, 24 SAY "Is " + STR(CURNT_YEAR,4) + " the correct year?"
  @ 13, 37 GET CONFIRM PICTURE "!"
  READ
  IF CONFIRM = "Y"
    STORE "Y" TO REPEAT
    USE ARCHIVE
    SET SAFETY OFF
    INDEX ON STATION + STR(YEAR) TO TEMP
    SET SAFETY ON
    LOCATE FOR YEAR = CURNT_YEAR
    IF .NOT. FOUND() AND EOF()
       CLOSE DATABASES
       DO ANNUAL
    ELSE
       @ 11,1 CLEAR TO 23,78
       SET COLOR TO GR+/*
       @ 10, 34 SAY "WARNING"
       SET COLOR TO GR+/R,W/R,R/N
       @ 12, 17 SAY "AN ANNUAL REPORT FOR "+STR(CURNT_YEAR,4)+" ALREADY EXISTS
       !"
```

```
      @ 14, 11 SAY "THE ANNUAL REPORT SHOULD BE COMPILED ONLY ONE TIME. IF
RUN"
      @ 15, 11 SAY "MORE THAN ONCE FOR A CALENDAR YEAR, THE DATABASE WILL BE"
      @ 16, 11 SAY "UPDATED WITH THE NEW INFORMATION. DO YOU WANT TO
CONTINUE?"
      @ 18, 37 GET CONTINUE PICTURE "Y"
      READ
      IF CONTINUE = "Y"
         DELETE ALL FOR YEAR = CURNT_YEAR
         PACK
         CLOSE DATABASES
         DO ANNUAL
       ELSE
         CLOSE DATABASES
       ENDIF
    ENDIF
  ENDIF
ENDDO WHILE REPEAT <> "Y"
RETURN
```

```
*********************************************************************************
* Program :      INIT.PRG                                                       *
* Author :       P. E. PAQUETTE                                                 *
* Date :         10/10/87                                                       *
* Purpose :      THIS MODULE SETS THE DBASE PARAMETERS WHICH WILL               *
*                DEFINE THE OPERATING ENVIRONMENT OF THE PROGRAM                *
* Input File  :  NONE                                                           *
* Output File :  NONE                                                           *
* Called By :    MAIN.PRG                                                       *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      NONE                                                           *
*   Global :     NONE                                                           *
*********************************************************************************

CLOSE DATABASES
RELEASE ALL EXCEPT USER*
*SET ESCAPE OFF WHEN PROGRAM IS DEBUGGED
SET SAFETY OFF
SET BELL OFF
SET STATUS OFF
SET COLOR TO GR+/R,W/R,N
SET SCOREBOARD OFF
SET TALK OFF
SET FUNCTION 2 TO ";"
SET FUNCTION 3 TO ";"
SET FUNCTION 4 TO ";"
SET FUNCTION 5 TO ";"
SET FUNCTION 6 TO ";"
SET FUNCTION 7 TO ";"
SET FUNCTION 8 TO ";"
SET FUNCTION 9 TO ";"
SET FUNCTION 10 TO ";"
RETURN
```

```
*********************************************************************************
* Program :      LOG.PRG                                                        *
* Author :       P. E. PAQUETTE                                                 *
* Date :         02/29/88                                                       *
* Purpose :      THIS MODULE RECORDS THE ACCESSES TO THE SYSTEM                 *
*                AS A MEANS OF PROVIDING AN AUDIT TRAIL OF                      *
*                TRANSACTIONS ON THE SYSTEM                                     *
* Input File :   ACCESS.DBF                                                     *
* Output File :  UPDATED ACCESS.DBF                                             *
* Called By :    MAIN.PRG                                                       *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      NONE                                                           *
*   Global :     NONE                                                           *
*********************************************************************************

use ACCESS
append blank
tempchoice = '      '
replace logtime with time()
replace logdate with date()
replace loguser with username
replace logunit with userunit
replace logchoice with choice
do case
     CASE CHOICE = 1
        tempchoice = 'UPDATE'
     CASE CHOICE = 2
        tempchoice = 'EDIT'
     CASE CHOICE = 3
        tempchoice = 'DELETE'
     CASE CHOICE = 4
         tempchoice = 'REPORTS'
     CASE CHOICE = 5
        tempchoice = 'QUIT'
endcase
replace logchoicec with tempchoice
close databases
return
```

```
*********************************************************************************
* Program :      MAIN.PRG                                                       *
* Author :       P. E. PAQUETTE                                                 *
* Date :         05/25/87                                                       *
* Purpose :      THIS MODULE PRESENTS THE USER WITH A MENU OF                   *
*                ALL OF THE REPORTS AVAILABLE.  IT CONTROLS                     *
*                THE EXECUTION OF ALL SUBORDINATE PROGRAMS                      *
* Input File :   NONE                                                           *
* Output File :  NONE                                                           *
* Called By :    SECURITY                                                       *
* Calls :        UPDATE, EDIT, DELETE, REPORTS, INIT                            *
* Variables                                                                     *
*   Local :      EXIT, CHOICE, DELAY                                            *
*   Global :     NONE                                                           *
*********************************************************************************

DO INIT
DO SECURITY
STORE 9 TO EXIT
DO WHILE EXIT > 5
  CLEAR
  @ 0,0 TO 24,79 DOUBLE
  @ 3,31 SAY "MARINE CORPS"
  @ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 6,32 SAY "MAIN MENU"
  @ 8,0 SAY CHR(204)
  @ 8,79 SAY CHR(185)
  @ 8,1 TO 8,78 DOUBLE
  @ 10,27 SAY [1.  UPDATE RECORDS]
  @ 11,27 SAY [2.  EDIT RECORDS]
  @ 12,27 SAY [3.  DELETE RECORDS]
  @ 13,27 SAY [4.  GENERATE REPORTS]
  @ 14,27 SAY [5.  QUIT]
  STORE 0 TO CHOICE
  @ 18,28 SAY "PLEASE ENTER A NUMBER"
  @ 18,54 GET CHOICE PICTURE "9"
  READ
  DO LOG
  DO CASE
    CASE CHOICE = 1
       DO UPDATE
    CASE CHOICE = 2
       DO EDIT
    CASE CHOICE = 3
       DO DELETE
    CASE CHOICE = 4
        DO REPORTS
    CASE CHOICE = 5
       QUIT
  OTHERWISE
    @ 10,17 CLEAR TO 19,55
    @ 10,17 TO 18,55 DOUBLE
    @ 14,25 SAY "ERROR IN INPUT VALUE"
    @ 15,25 SAY "ENTER A NUMBER LISTED"
    ?? CHR(7)
```

```
    STORE 1 TO DELAY
    DO WHILE DELAY < 35
      STORE DELAY + 1 TO DELAY
    ENDDO WHILE DELAY <35
  ENDCASE
ENDDO WHILE EXIT > 5
```

```
*********************************************************************************
* Program :      MONTHLY                                                        *
* Author :       P. E. PAQUETTE                                                 *
* Date :         12/19/87                                                       *
* Purpose :      THIS PROGRAM ALLOWS THE USER TO VEIW ANY MONTHLY               *
*                REPORT STORED IN THE DATABASE FILES. IT ALSO                   *
*                ALLOWS THE USER TO GET A HARDCOPY OF THE REPORT                *
*                IF DESIRED.                                                    *
* Input File  :  DBF (as chosen by the user)                                    *
* Output File :  NONE                                                           *
* Called By :    REPORTS                                                        *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      MURDERCNT,MANCOUNT,RAPECOUNT,ARAPECOUNT                        *
*                ARMEDCOUNT,OTHERCOUNT,ASSLTCOUNT,OTWPNCOUNT                    *
*                SIMPLCOUNT,ENTRYCOUNT,TRYCOUNT,DODCOUNT                        *
*                NDODCOUNT,GOVTHEFT,CIVILTHEFT,AUTOCOUNT                        *
*                VANCOUNT,GAUTOCOUNT,NAUTOCOUNT,NARCCOUNT                       *
*                SELLCOUNT,SMOKECOUNT,DEALCOUNT,DWIBASE,DWI                     *
*                DUIBASE,DUI,TINDXTOTAL,CHOICE,STATION,DBF,EXIT                 *
*                ANSWER,MONTH,VERIFY,DELAY,HOMICIDE,FORCERAPE                   *
*                ROBBERY,ASSAULT,BURGLARY,LARCENY,AUTOTHEFT                     *
*                NDEXTOTAL,VIOLENT,PROPCRIME,DRUGTOTAL,DUITOTAL                 *
*                DWITOTAL,ALCHLTOTAL,CENTER                                     *
*   Global :     NONE                                                           *
*********************************************************************************

STORE 0 TO
MURDERCNT,MANCOUNT,RAPECOUNT,ARAPECOUNT,ARMEDCOUNT,OTHERCOUNT
STORE 0 TO
ASSLTCOUNT,OTWPNCOUNT,SIMPLCOUNT,ENTRYCOUNT,TRYCOUNT,DODCOUNT
STORE 0 TO NDODCOUNT,GOVTHEFT,CIVILTHEFT,AUTOCOUNT,VANCOUNT,GAUTOCOUNT
STORE 0 TO NAUTOCOUNT,NARCCOUNT,SELLCOUNT,SMOKECOUNT,DEALCOUNT
STORE 0 TO DWIBASE,DWI,DUIBASE,DUI,TINDXTOTAL
STORE 99 TO CHOICE
STORE "              " TO STATION
STORE "       " TO DBF
STORE "N" TO EXIT
STORE "Y" TO VERIFY
STORE "N" TO ANSWER
STORE 23 TO MONTH


DO WHILE EXIT = "N"
  DO WHILE CHOICE > 20
   CLEAR
   @ 0,0 TO 24,79 DOUBLE
   @ 3,31 SAY "MARINE CORPS"
   @ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
   @ 6,20 SAY "      INSTALLATION MENU"
   @ 8,0 SAY CHR(204)
   @ 8,79 SAY CHR(185)
   @ 8,1 TO 8,78 DOUBLE
   @ 10,15 SAY "1. MCLB ALBANY          11. MCB CAMP LEJEUNE"
   @ 11,15 SAY "2. MCLB BARSTOW         12. MCRD PARRIS ISLAND"
```

```
@ 12,15 SAY "3. MCAS BEAUFORT          13. MCB CAMP PENDLETON"
@ 13,15 SAY "4. CAMP BUTLER           14. MCDEC QUANTICO"
@ 14,15 SAY "5. MCAS CHERRY POINT      15. MCRD SAN DIEGO"
@ 15,15 SAY "6. MCAS EL TORO          16. MCAS TUSTIN"
@ 16,15 SAY "7. CAMP ELMORE           17. MCAGCC 29 PALMS"
@ 17,15 SAY "8. HENDERSON HALL         18. MCAS YUMA"
@ 18,15 SAY "9. MCAS IWAKUNI          19. CAMP SMITH"
@ 19,14 SAY "10. MCAS KANEOHE          20. MCAS NEW RIVER"
@ 21,18 SAY "SELECT THE STATION YOU WISH TO VIEW"
@  22,24 SAY "PLEASE ENTER A NUMBER"
@ 22,46 GET CHOICE PICTURE "99"
READ
DO CASE
  CASE CHOICE =1
      STORE "MCLB ALBANY" TO STATION
      STORE "ALBANY" TO DBF
  CASE CHOICE =2
      STORE "MCLB BARSTOW" TO STATION
      STORE "BARSTOW" TO DBF
  CASE CHOICE =3
      STORE "MCAS BEAUFORT" TO STATION
      STORE "BEAUFORT" TO DBF
  CASE CHOICE =4
      STORE "CAMP BUTLER" TO STATION
      STORE "BUTLER" TO DBF
  CASE CHOICE =5
      STORE "CHERRY POINT" TO STATION
      STORE "CHERRY" TO DBF
  CASE CHOICE =6
      STORE "MCAS EL TORO" TO STATION
      STORE "ELTORO" TO DBF
  CASE CHOICE =7
      STORE "CAMP ELMORE" TO STATION
      STORE "ELMORE" TO DBF
  CASE CHOICE =8
      STORE "HENDERSON HALL" TO STATION
      STORE "HENDRSON" TO DBF
  CASE CHOICE =9
      STORE "MCAS IWAKUNI" TO STATION
      STORE "IWAKUNI" TO DBF
  CASE CHOICE =10
      STORE "MCAS KANEOHE" TO STATION
      STORE "KANEOHE" TO DBF
  CASE CHOICE =11
      STORE "MCB CAMP LEJEUNE" TO STATION
      STORE "LEJEUNE" TO DBF
  CASE CHOICE =12
      STORE "MCRD PARRIS ISLAND" TO STATION
      STORE "PI" TO DBF
  CASE CHOICE =13
      STORE "MCB CAMP PENDLETON" TO STATION
      STORE "PENDELTN" TO DBF
  CASE CHOICE =14
      STORE "MCDEC QUANTICO" TO STATION
      STORE "QUANTICO" TO DBF
```

```
  CASE CHOICE =15
      STORE "MCRD SAN DIEGO" TO STATION
      STORE "SANDIEGO" TO DBF
  CASE CHOICE =16
      STORE "MCAS TUSTIN" TO STATION
      STORE "TUSTIN" TO DBF
  CASE CHOICE =17
      STORE "MCAGCC TWENTYNINE PALMS" TO STATION
      STORE "STUMPS" TO DBF
  CASE CHOICE =18
      STORE "MCAS YUMA" TO STATION
      STORE "YUMA" TO DBF
  CASE CHOICE =19
      STORE "CAMP SMITH" TO STATION
      STORE "SMITH" TO DBF
  CASE CHOICE =20
      STORE "MCAS NEW RIVER" TO STATION
      STORE "NEWRIVER" TO DBF
  OTHERWISE
      @ 10,17 CLEAR TO 19,55
      @ 10,17 TO 18,55 DOUBLE
      @ 14,25 SAY "ERROR IN INPUT VALUE"
      @ 15,25 SAY "ENTER A NUMBER LISTED"
      ?? CHR(7)
      STORE 1 TO DELAY
      DO WHILE DELAY < 50
        STORE DELAY +1 TO DELAY
      ENDDO WHILE DELAY < 50
      CHOICE = 99
  ENDCASE
ENDDO WHILE CHOICE > 20
DO WHILE (MONTH < 1) .OR. (MONTH > 12)
  STORE  "MM/YY"   TO DATE
  @ 10,01 CLEAR TO 23,78
  @ 14,25 SAY "WHAT IS THE DATE OF THE"
  @ 15,25 SAY "REPORT TO BE COMPILED ?"
  @ 16,34 GET DATE PICTURE "99/99"
  READ
  MONTH = VAL(SUBSTR(DATE,1,2))
  IF (MONTH < 1) .OR. (MONTH > 12)
    @ 10,01 CLEAR TO 23,78
    @ 14,21 SAY "THERE ARE ONLY 12 MONTHS IN A YEAR !"
    STORE 1 TO DELAY
    DO WHILE DELAY < 50
        STORE DELAY +1 TO DELAY
    ENDDO WHILE DELAY < 50
  ENDIF
ENDDO WHILE MONTH
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,31 SAY "MARINE CORPS"
@ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 6,28 SAY "VALIDATION OF INPUT"
@ 8,0 SAY CHR(204)
@ 8,1 TO 8,78 DOUBLE
```

```
  @ 8,79 SAY CHR(185)
  @ 10,7 SAY "THE INFORMATION WHICH YOU HAVE PROVIDED WILL BE USED TO
SEARCH"
  @ 11,7 SAY "THE CRIME REPORTING STATISTICS DATA BASE.  PLEASE CHECK THE"
  @ 12,7 SAY "INFORMATION BELOW TO ENSURE THAT YOUR ENTRIES ARE CORRECT"
  @ 16,26 SAY "INSTALLATION NAME:  &STATION"
  @ 18,26 SAY "DATE OF REPORT: &DATE "
  @ 22,26 SAY "ARE ALL ENTRIES CORRECT?"
  STORE "Y" TO VERIFY
  @ 23,37 GET VERIFY PICTURE "!"
  READ
  IF VERIFY = "Y"
    STORE "Y" TO EXIT
  ELSE
    STORE "N" TO EXIT
    STORE 99 TO CHOICE
  ENDIF
ENDDO WHILE EXIT = N


@ 10,2 CLEAR TO 23,78
SET COLOR TO GR+/*
@ 14,34 SAY "Thinking....."
SET COLOR TO GR+/R,W/R,N
SELECT B
USE &DBF
GO TOP


DO WHILE .NOT. EOF()
   if substr(DATE,4,2) = substr("&DATE",4,2)
    DO CASE
     CASE (CRIME = "MURDER")
         MURDERCNT = F1 + M->MURDERCNT
     CASE (CRIME = "MANSLAUGHTER")
         MANCOUNT = F1 + MANCOUNT
     CASE (CRIME = "RAPE")
         RAPECOUNT = F1 + RAPECOUNT
     CASE (CRIME = "ATTEMPTED RAPE")
         ARAPECOUNT = F1 + ARAPECOUNT
     CASE (CRIME = "ROBBERY W/FIRE ARM")
         ARMEDCOUNT = F1 + ARMEDCOUNT
     CASE (CRIME = "ROBBERY W/OTHER WEAPON")
         OTHERCOUNT = F1 + OTHERCOUNT
     CASE (CRIME = "ASSAULT W/FIRE ARM")
         ASSLTCOUNT = F1 + ASSLTCOUNT
     CASE (CRIME = "ASSAULT W/OTHER WEAPON")
         OTWPNCOUNT = F1 + OTWPNCOUNT
     CASE (CRIME = "SIMPLE ASSAULT")
         SIMPLCOUNT = F1 + SIMPLCOUNT
     CASE (CRIME = "ENTRY")
         ENTRYCOUNT = F1 + ENTRYCOUNT
     CASE (CRIME = "ATTEMPTED ENTRY")
         TRYCOUNT = F1 + TRYCOUNT
     CASE (CRIME = "GOVT. PROPERTY")
```

```
        DODCOUNT = F1 + DODCOUNT
    CASE (CRIME = "NON-GOVT. PROPERTY")
        NDODCOUNT = F1 + NDODCOUNT
    CASE (CRIME = "LARCENY GOVT. PROPERTY")
        GOVTHEFT = F1 + GOVTHEFT
    CASE (CRIME = "LARCENY NON-GOVT. PROPERTY")
        CIVILTHEFT = F1 + CIVILTHEFT
    CASE (CRIME = "AUTO THEFT")
        AUTOCOUNT = F1 + AUTOCOUNT
    CASE (CRIME = "OTHER VEHICLE THEFT")
        VANCOUNT = F1 + VANCOUNT
    CASE (CRIME = "GOVT. VEHICLE")
        GAUTOCOUNT = F1 + GAUTOCOUNT
    CASE (CRIME = "NON-GOVT. VEHICLE")
        NAUTOCOUNT = F1 + NAUTOCOUNT
    CASE (CRIME = "NARCOTICS USE & POSSESSION")
        NARCCOUNT = F1 + NARCCOUNT
    CASE (CRIME = "NARCOTICS SALE & TRAFFICKING")
        SELLCOUNT = F1 + SELLCOUNT
    CASE (CRIME = "MARIJUANA USE & POSSESSION")
        SMOKECOUNT = F1 + SMOKECOUNT
    CASE (CRIME = "MAJIJUANA SALE & TRAFFICKING")
        DEALCOUNT = F1 + DEALCOUNT
    CASE (CRIME = "DWI ON BASE")
        DWIBASE = F1 + DWIBASE
    CASE (CRIME = "DWI OFF BASE")
        DWI = F1 + DWI
    CASE (CRIME = "DUI ON BASE")
        DUIBASE = F1 + DUIBASE
    CASE (CRIME = "DUI OFF BASE")
        DUI = F1 + DUI
    ENDCASE
  else
   ?? "could not locate record"
  endif
   skip
ENDDO WHILE NOT EOF


store MURDERCNT + MANCOUNT to HOMICIDES
store RAPECOUNT + ARAPECOUNT to FORCERAPE
store ARMEDCOUNT + OTHERCOUNT to ROBBERY
store ASSLTCOUNT + OTWPNCOUNT to ASSAULT
store ENTRYCOUNT + TRYCOUNT + DODCOUNT + NDODCOUNT to BURGLARY
store GOVTHEFT + CIVILTHEFT to LARCENY
store AUTOCOUNT + VANCOUNT + GAUTOCOUNT + NAUTOCOUNT to AUTOTHEFT
store HOMICIDES + FORCERAPE + ROBBERY + ASSAULT + BURGLARY + LARCENY ;
          + AUTOTHEFT to INDEXTOTAL
store homicides + forcerape + robbery + assault to violent
store burglary + larceny + autotheft to propcrime
store narccount  + sellcount + smokecount + dealcount to drugtotal
store duibase + dui to duitotal
store dwibase + dwi to dwitotal
store duitotal + dwitotal to alcholtotl
CLEAR
```

```
STORE (80-LEN("&STATION"))/2 TO CENTER
@ 0,0 TO 24,79 DOUBLE
@ 3,CENTER SAY "&STATION"
@ 4,32 SAY "MONTHLY SUMMARY"
@ 5,38 SAY "FOR"
@ 6,37 SAY "&DATE"
@ 8,0 SAY CHR(204)
@ 8,79 SAY CHR(185)
@ 8,1 TO 8,78 DOUBLE
@ 10, 5  SAY "I.  INDEX CRIMES"
@ 11, 5  SAY "    A.  CRIMES AGAINST PERSONS"
@ 12, 5  SAY "        1.  MURDER"
@ 12, 45 SAY HOMICIDES
@ 13, 5  SAY "        2.  RAPE"
@ 13, 45 SAY FORCERAPE
@ 14, 5  SAY "        3.  ROBBERY"
@ 14, 45 SAY ROBBERY
@ 15, 5  SAY "        4.  AGGREVATED ASSUALT"
@ 15, 45 SAY ASSAULT
@ 17, 5  SAY "    B.  CRIMES AGAINST PROPERTY"
@ 18, 5  SAY "        1.  BURGLARY/HOUSEBREAKING"
@ 18, 45 SAY BURGLARY
@ 19, 5  SAY "        2.  LARCENY"
@ 19, 45 SAY LARCENY
@ 20, 5  SAY "        3.  MOTOR VEHICLE THEFT"
@ 20, 45 SAY AUTOTHEFT
@ 23, 25 SAY "PRESS ANY KEY TO CONTINUE"
READ
@ 10, 2  CLEAR TO 23,78
@ 10, 5  SAY "II.  DRUG RELATED OFFENSES"
@ 11, 5  SAY "     A. NARCOTICS/DANGEROUS DRUGS"
@ 12, 5  SAY "        1.  Use & Possession"
@ 12, 45  SAY NARCCOUNT
@ 13, 5  SAY "        2.  Sale & Trafficking"
@ 13, 45  SAY SELLCOUNT
@ 15, 5  SAY "     B. MARIJUANA"
@ 16, 5  SAY "        1.  Use & Possession"
@ 16, 45  SAY SMOKECOUNT
@ 17, 5  SAY "        2.  Sale & Trafficking"
@ 17, 45  SAY DEALCOUNT
@ 23, 25  SAY "PRESS ANY KEY TO CONTINUE"
READ
@ 10,2 CLEAR TO 23,78
@ 10,5  SAY "III. ALCOHOL RELATED OFFENSES"
@ 11,5  SAY "     A.  DWI (+.10 BAC)"
@ 12,5  SAY "        1.  DWI ON BASE"
@ 12,45 SAY DWIBASE
@ 13,5  SAY "        2.  DWI OFF BASE"
@ 13,45 SAY DWI
@ 15,5  SAY "     B.  DUI (.05 -.09 BAC)"
@ 16,5  SAY "        1.  DUI ON BASE"
@ 16,45 SAY DUIBASE
@ 17,5  SAY "        2.  DUI OFF BASE"
@ 17,45 SAY DUI
@ 22,20 SAY "WOULD YOU LIKE A PRINTOUT OF THIS REPORT ?"
```

```
@ 23,39 GET ANSWER PICTURE "!"
READ


IF ANSWER <> "Y"
  CLOSE DATABASES
  RELEASE ALL EXCEPT USER*
ELSE
  @ 10,2  CLEAR TO 23,78
  ? CHR(7)
  @ 14,27  SAY "PREPARE TO PRINTER TO PRINT"
  wait
  @ 0,0
  SET DEVICE TO PRINT
  @ 3,31   SAY "MARINE CORPS"
  @ 4,20   SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 5,20   SAY "        MONTHLY SUMMARY"
  @ 8,5    SAY "INSTALLATION: "+"&STATION"
  @ 8,49   SAY "DATE OF REPORT: "+"&DATE"
  @ 11, 5  SAY "I.  INDEX CRIMES"
  @ 12, 5  SAY "    A.  CRIMES AGAINST PERSONS"
  @ 13, 5  SAY "        1.  MURDER"
  @ 13, 45 SAY HOMICIDES
  @ 14, 5  SAY "        2.  RAPE"
  @ 14, 45 SAY FORCERAPE
  @ 15, 5  SAY "        3.  ROBBERY"
  @ 15, 45 SAY ROBBERY
  @ 16, 5  SAY "        4.  AGGREVATED ASSUALT"
  @ 16, 45 SAY ASSAULT
  @ 17, 5  SAY "    B.  CRIMES AGAINST PROPERTY"
  @ 18, 5  SAY "        1.  BURGLARY/HOUSEBREAKING"
  @ 18, 45 SAY BURGLARY
  @ 19, 5  SAY "        2.  LARCENY"
  @ 19, 45 SAY LARCENY
  @ 20, 5  SAY "        3.  MOTOR VEHICLE THEFT"
  @ 20, 45 SAY AUTOTHEFT
  @ 22, 4  SAY "II.  DRUG RELATED OFFENSES"
  @ 23, 5  SAY "    A.  NARCOTICS/DANGEROUS DRUGS"
  @ 24, 5  SAY "        1.  Use & Possession"
  @ 24, 45  SAY NARCCOUNT
  @ 25, 5  SAY "        2.  Sale & Trafficking"
  @ 25, 45  SAY SELLCOUNT
  @ 26, 5  SAY "    B.  MARIJUANA"
  @ 27, 5  SAY "        1.  Use & Possession"
  @ 27, 45  SAY SMOKECOUNT
  @ 28, 5  SAY "        2.  Sale & Trafficking"
  @ 28, 45  SAY DEALCOUNT
  @ 30,3  SAY "III.  ALCOHOL RELATED OFFENSES"
  @ 31,3  SAY "     A.  DWI (+.10 BAC)"
  @ 32,3  SAY "         1.  DWI ON BASE"
  @ 32,45 SAY DWIBASE
  @ 33,3  SAY "         2.  DWI OFF BASE"
  @ 33,45 SAY DWI
  @ 34,3  SAY "     B.  DUI (.05 -.09 BAC)"
  @ 35,3  SAY "         1.  DUI ON BASE"
```

```
  @ 35,45 SAY DUIBASE
  @ 36,3  SAY "         2.  DUI OFF BASE"
  @ 36,45 SAY DUI
  @ 41,9  SAY "TOTAL INDEX CRIMES"
  @ 41,45 SAY INDEXTOTAL
  @ 42,9  SAY "TOTAL DRUG RELATED OFFENSES"
  @ 42,45 SAY DRUGTOTAL
  @ 43,9  SAY "TOTAL ALCOHOL RELATED DRIVING OFFENSES"
  @ 43,45 SAY ALCHOLTOTL
  EJECT
  SET DEVICE TO SCREEN
  CLOSE DATABASES
  RELEASE ALL EXCEPT USER*
ENDIF
RETURN
```

```
*********************************************************************************
* Program :      MAIN PRINT PROGRAM (PRINT.PRG)                                 *
* Author :       P. E. PAQUETTE                                                 *
* Date :         1/8/88                                                         *
* Purpose :      THIS PROGRAM CONTROLS ALL OF THE PRINT MODULES                 *
*                WHICH PRODUCE THE YEARLY REPORT.                               *
* Input File  :  NONE                                                           *
* Output File :  NONE                                                           *
* Called By :    MAIN.PRG                                                       *
* Calls :        PRINT1, PRINT2, PRINT3, PRINT4                                 *
* Variables                                                                     *
*   Local :      ANS, CURNT_YEAR, LAST_YEAR, CONFIRM, REPEAT                    *
*   Global :     NONE                                                           *
*********************************************************************************

CLEAR
STORE "N" TO ANS,CONFIRM,REPEAT
STORE 1987 TO CURNT_YEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,31 SAY "MARINE CORPS"
@ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 6,20 SAY "        REPORTS GENERATION"
@ 8,0 SAY CHR(204)
@ 8,1 TO 8,78 DOUBLE
@ 8,79 SAY CHR(185)
@ 10, 5 SAY "     This program is designed for use with a standard dot matrix"
@ 11, 5 SAY "printer. It is configured to use continuous form feed paper with"
@ 12, 5 SAY "at least 80 columns.  Please ensure that the printer is ready to"
@ 13, 5 SAY "go."
@ 15, 17  SAY "Are you ready to proceed with the print job?"
@ 16, 37 GET ANS PICTURE "!"
READ
IF ANS <> "Y"
  RETURN
ELSE
  DO WHILE REPEAT <> "Y"
    @ 10, 3 CLEAR TO 20,78
    @ 12, 21 SAY "What is the year of this annual report?"
    @ 13, 36 GET CURNT_YEAR PICTURE "9999"
    READ
    STORE (CURNT_YEAR - 1) TO LAST_YEAR
    @ 10, 3 CLEAR TO 20, 78
    @ 12, 24 SAY "Is " + STR(CURNT_YEAR,4) + " the correct year?"
    @ 13, 37 GET CONFIRM PICTURE "!"
    READ
    IF CONFIRM = "Y"
      STORE "Y" TO REPEAT
      USE ARCHIVE
      INDEX ON STATION + STR(YEAR) TO TEMP
      LOCATE FOR YEAR = CURNT_YEAR
      IF .NOT. FOUND() .AND. EOF()
        @ 9, 2 CLEAR TO 23,78
        @ 12,28 SAY "NO REPORT EXISTS FOR " + STR(CURNT_YEAR,4)
        @ 13,18 SAY "TRY THE `COMPILE' OPTION IN THE REPORTS MENU"
      ELSE
```

```
            DO PRINT1
            DO PRINT2
            DO PRINT3
            DO PRINT4
        ENDIF NOT FOUND
      ENDIF CONFIRM <> "Y"
    ENDDO WHILE REPEAT <> "Y"
ENDIF ANS <> "Y"
ERASE TEMP
CLOSE DATABASES
RELEASE ALL EXCEPT USER*
RETURN
```

```
*********************************************************************************
* Program :      FIRST PRINT MODULE  (PRINT1.PRG)                               *
* Author :       P. E. PAQUETTE                                                 *
* Date :         1/8/88                                                         *
* Purpose :      THIS MODULE WILL PRINT ALL THE CRIME                           *
*                NFORMATION IN THE TWO COLUMN FORMAT AS WELL                    *
*                AS CALCULATING THE TOTALS AND PERCENT CHANGE                   *
*                BETWEEN THE YEARS                                              *
* Input File :   ARCHIVE.DBF                                                    *
* Output File :  NONE                                                           *
* Called By :    PRINT                                                          *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      TEMP_YEAR, LAST_YEAR, HEADING, YR_TOTAL,                       *
*                LYR_TOTAL, COUNT, PLACE, CRIME, CENTER, PLACE                  *
*                STATION, TEMP1, TEMP2, PERCNTCHG                               *
*   Global ;     NONE                                                           *
*********************************************************************************

@ 10,3 clear to 22,78
SET COLOR TO GR+/*
@ 12,24 SAY "Printing....."
SET COLOR TO GR+/R,W/R,N
STORE SPACE(62) TO HEADING
STORE 0 TO YR_TOTAL,LYR_TOTAL
STORE 1 TO COUNT,PLACE
SET SAFETY OFF
SET DEVICE TO PRINT
DO WHILE COUNT < 15
   DO CASE
    CASE COUNT =1
        STORE "INDEX CRIMES REPORTED TO HQMC" TO HEADING
        STORE "INDEXTOTAL" TO CRIME
    CASE COUNT =2
        STORE "HOMICIDES REPORTED TO HQMC" TO HEADING
        STORE "HOMICIDES" TO CRIME
    CASE COUNT =3
        STORE "FORCIBLE RAPES REPORTED TO HQMC" TO HEADING
        STORE "FORCERAPE" TO CRIME
    CASE COUNT =4
        STORE "ROBBERY CRIMES REPORTED TO HQMC" TO HEADING
        STORE "ROBBERY" TO CRIME
    CASE COUNT =5
        STORE "AGGRAVATED ASSAULTS REPORTED TO HQMC" TO HEADING
        STORE "ASSAULT" TO CRIME
    CASE COUNT =6
        STORE "CRIMES AGAINST PROPERTY REPORTED TO HQMC" TO HEADING
        STORE "PROPCRIME" TO CRIME
    CASE COUNT =7
        STORE "BURGLARY COMPLAINTS REPORTED TO HQMC" TO HEADING
        STORE "BURGLARY" TO CRIME
    CASE COUNT =8
        STORE "LARCENY COMPLAINTS REPORTED TO HQMC" TO HEADING
        STORE "LARCENY" TO CRIME
    CASE COUNT =9
```

```
      STORE "MOTOR VEHICLE THEFTS REPORTED TO HQMC" TO HEADING
      STORE "AUTOTHEFT" TO CRIME
  CASE COUNT =10
      STORE "DRUG RELATED OFFENSES REPORTED TO HQMC" TO HEADING
      STORE "DRUGTOTAL" TO CRIME
  CASE COUNT =11
      STORE "NARCOTICS & DANGEROUS DRUG TRAFFICKING OFFENSES" + ;
          " REPORTED TO HQMC" TO HEADING
      STORE "SELLCOUNT" TO CRIME
  CASE COUNT =12
      STORE "NARCOTICS & DANGEROUS DRUG POSSESSION OFFENSES" + ;
          " REPORTED TO HQMC" TO HEADING
      STORE "NARCCOUNT" TO CRIME
  CASE COUNT =13
      STORE "MARIJUANA TRAFFICKING OFFENSES REPORTED TO HQMC" TO HEADING
      STORE "DEALCOUNT" TO CRIME
  CASE COUNT =14
      STORE "MARIJUANA POSSESSION OFFENSES REPORTED TO HQMC" TO HEADING
      STORE "SMOKECOUNT" TO CRIME
ENDCASE
STORE (80-LEN(TRIM("&HEADING")))/2 TO CENTER
@ 5, CENTER SAY "&HEADING"
@ 8, 42 SAY STR(LAST_YEAR,4)
@ 8, 52 SAY STR(CURNT_YEAR,4)
@ 8, 60 SAY "% OF CHANGE"
STORE 10 TO LINE
STORE 1 TO PACE
DO WHILE PLACE < 21
  DO CASE
    CASE PLACE =1
     STORE "MCLB ALBANY" TO STATION
    CASE PLACE =2
     STORE "MCLB BARSTOW" TO STATION
    CASE PLACE =3
     STORE "MCAS BEAUFORT" TO STATION
    CASE PLACE =4
     STORE "CAMP BUTLER" TO STATION
    CASE PLACE =5
     STORE "CHERRY POINT" TO STATION
    CASE PLACE =6
     STORE "MCAS EL TORO" TO STATION
    CASE PLACE =7
     STORE "CAMP ELMORE" TO STATION
    CASE PLACE =8
     STORE "HENDERSON HALL" TO STATION
    CASE PLACE =9
     STORE "MCAS IWAKUNI" TO STATION
    CASE PLACE =10
     STORE "MCAS KANEOHE" TO STATION
    CASE PLACE =11
     STORE "MCB CAMP LEJEUNE" TO STATION
    CASE PLACE =12
     STORE "MCRD PARRIS ISLAND" TO STATION
    CASE PLACE =13
     STORE "MCB CAMP PENDLETON" TO STATION
```

```
        CASE PLACE =14
          STORE "MCDEC QUANTICO" TO STATION
        CASE PLACE =15
          STORE "MCRD SAN DIEGO" TO STATION
        CASE PLACE =16
          STORE "MCAS TUSTIN" TO STATION
        CASE PLACE =17
          STORE "MCAGCC TWENTYNINE PALMS" TO STATION
        CASE PLACE =18
          STORE "MCAS YUMA" TO STATION
        CASE PLACE =19
          STORE "CAMP SMITH" TO STATION
        CASE PLACE =20
          STORE "MCAS NEW RIVER" TO STATION
      ENDCASE
      GO TOP
      SET EXACT OFF      -
      LOCATE FOR (STATION =  "&STATION") .AND. (YEAR = LAST_YEAR)
      @ LINE, 5  SAY "&STATION"
      @ LINE, 42 SAY &CRIME
      STORE &CRIME TO TEMP1
      STORE &CRIME + LYR_TOTAL TO LYR_TOTAL
      GO TOP
      LOCATE FOR (STATION = "&STATION") .AND. (YEAR = CURNT_YEAR)
      SET EXACT ON
      @ LINE, 52 SAY &CRIME
      STORE &CRIME TO TEMP2
      STORE &CRIME + YR_TOTAL TO YR_TOTAL
      IF TEMP1 = 0
        STORE TEMP2 * 100 TO PERCNTCHG
      ELSE
        STORE ROUND((((TEMP2-TEMP1)/(TEMP1))*100),2) TO PERCNTCHG
      ENDIF
      @ LINE, 58 SAY PERCNTCHG
      STORE PLACE + 1 TO PLACE
      STORE LINE + 2 TO LINE
    ENDDO WHILE PLACE < 21
  STORE LINE + 1 TO LINE
  @ LINE, 5 SAY "TOTAL"
  @ LINE, 36 SAY LYR_TOTAL
  @ LINE, 46 SAY YR_TOTAL
  IF LYR_TOTAL = 0
    STORE YR_TOTAL * 100 TO PERCNTCHG
  ELSE
    STORE ROUND((((YR_TOTAL-LYR_TOTAL)/LYR_TOTAL)*100),2) TO PERCNTCHG
  ENDIF
  @ LINE, 58 SAY PERCNTCHG
  STORE COUNT + 1 TO COUNT
ENDDO WHILE COUNT < 15
SET SAFETY ON
EJECT
SET DEVICE TO SCREEN
RELEASE ALL EXCEPT USER*
RETURN
```

```
*********************************************************************************
* Program :      SECOND PRINT MODULE  (PRINT2.PRG)                              *
* Author :       P. E. PAQUETTE                                                 *
* Date :         12/28/87                                                       *
* Purpose :      THIS MODULE WILL PRINT ALL THE DUI & DWI                       *
*                INFORMATION EXTRACTED FROM THE INSTALLATION                    *
*                DATABASES.  ITS FUNCTIONING IS SIMILAR TO THAT                 *
*                OF PRINT MODULE NUMBER ONE                                     *
* Input File :   ARCHIVE.DBF                                                    *
* Output File :  NONE                                                           *
* Called By :    PRINT.PRG                                                      *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      HEADING, YR_TOTAL, LYR_TOTAL, COUNT, PLACE                     *
*                CRIME1, CRIME2, CENTER, LINE, PLACE, TEMP1TOTAL                *
*                TEMP2TOTAL, TEMP3TOTAL, TEMP4TOTAL, TEMP5TOTAL,                *
*                TEMP6TOTAL, STATION, LAST_TOTAL, CRNT_TOTAL,                   *
*                PERCNTCHG                                                      *
*   Global :     NONE                                                           *
*********************************************************************************

@ 10, 3  CLEAR TO 22, 78
SET COLOR TO GR+/*
@ 12, 24   SAY "Printing....."
SET COLOR TO GR+/R,W/R,N
STORE SPACE(62) TO HEADING
STORE 0 TO YR_TOTAL,LYR_TOTAL
STORE 1 TO COUNT,PLACE
@ 0,0
SET DEVICE TO PRINT
DO WHILE COUNT < 3
   DO CASE
    CASE COUNT =1
        STORE "DRIVING VIOLATIONS RELATED TO ALCOHOL ABUSE" TO HEADING
        STORE "DWIBASE" TO CRIME1
        STORE "DWI" TO CRIME2
    CASE COUNT =2
        STORE "DUIBASE" TO CRIME1
        STORE "DUI" TO CRIME2
   ENDCASE
   STORE (80-LEN(TRIM("&HEADING")))/2 TO CENTER
   @ 4, CENTER SAY "&HEADING"
   @ 7, 20 SAY STR(LAST_YEAR,4)
   @ 7, 29 SAY STR(LAST_YEAR,4)
   @ 7, 38 SAY STR(CURNT_YEAR,4)
   @ 7, 47 SAY STR(CURNT_YEAR,4)
   @ 7, 56 SAY STR(LAST_YEAR,4)
   @ 7, 65 SAY STR(CURNT_YEAR,4)
   @ 7, 74 SAY "%"
   @ 8, 18 SAY "ON BASE  OFF BASE  ON BASE  OFF BASE  TOTAL   TOTAL"
   @ 8, 72 SAY "CHANGE"
   @ 9, 21 SAY "&CRIME2"
   @ 9, 30 SAY "&CRIME2"
   @ 9, 39 SAY "&CRIME2"
   @ 9, 48 SAY "&CRIME2"
```

```
@ 9, 57 SAY "&CRIME2"
@ 9, 66 SAY "&CRIME2"
STORE 11 TO LINE
STORE 1 TO PLACE
STORE 0 TO TEMP1TOTAL,TEMP2TOTAL,TEMP3TOTAL
STORE 0 TO TEMP4TOTAL,TEMP5TOTAL,TEMP6TOTAL
DO WHILE PLACE < 21
  DO CASE
    CASE PLACE =1
     STORE "MCLB ALBANY" TO STATION
    CASE PLACE =2
     STORE "MCLB BARSTOW" TO STATION
    CASE PLACE =3
     STORE "MCAS BEAUFORT" TO STATION
    CASE PLACE =4
     STORE "CAMP BUTLER" TO STATION
    CASE PLACE =5
     STORE "CHERRY POINT" TO STATION
    CASE PLACE =6
     STORE "MCAS EL TORO" TO STATION
    CASE PLACE =7
     STORE "CAMP ELMORE" TO STATION
    CASE PLACE =8
     STORE "HENDERSON HALL" TO STATION
    CASE PLACE =9
     STORE "MCAS IWAKUNI" TO STATION
    CASE PLACE =10
     STORE "MCAS KANEOHE" TO STATION
    CASE PLACE =11
     STORE "MCB CAMP LEJEUNE" TO STATION
    CASE PLACE =12
     STORE "MCRD PARRIS ISLAND" TO STATION
    CASE PLACE =13
     STORE "MCB CAMP PENDLETON" TO STATION
    CASE PLACE =14
     STORE "MCDEC QUANTICO" TO STATION
    CASE PLACE =15
     STORE "MCRD SAN DIEGO" TO STATION
    CASE PLACE =16
     STORE "MCAS TUSTIN" TO STATION
    CASE PLACE =17
     STORE "MCAGCC 29 PALMS" TO STATION
    CASE PLACE =18
     STORE "MCAS YUMA" TO STATION
    CASE PLACE =19
     STORE "CAMP SMITH" TO STATION
    CASE PLACE =20
     STORE "MCAS NEW RIVER" TO STATION
  ENDCASE
  GO TOP
  SET EXACT OFF
 LOCATE FOR (STATION =  "&STATION") .AND. (YEAR = LAST_YEAR)
 @ LINE, 1    SAY "&STATION"
 @ LINE, 20   SAY STR(&CRIME1,4)
 @ LINE, 29   SAY STR(&CRIME2,4)
```

```
      STORE &CRIME1 + &CRIME2 TO LAST_TOTAL
      STORE &CRIME1 + TEMP1TOTAL TO TEMP1TOTAL
      STORE &CRIME2 + TEMP2TOTAL TO TEMP2TOTAL
      STORE LAST_TOTAL  + TEMP5TOTAL TO TEMP5TOTAL
      GO TOP
      LOCATE FOR (STATION = "&STATION") .AND. (YEAR = CURNT_YEAR)
      SET EXACT ON
      @ LINE, 38   SAY STR(&CRIME1,4)
      @ LINE, 47   SAY STR(&CRIME2,4)
      STORE &CRIME1 + &CRIME2 TO CRNT_TOTAL
      STORE &CRIME1 + TEMP3TOTAL TO TEMP3TOTAL
      STORE &CRIME2 + TEMP4TOTAL TO TEMP4TOTAL
      STORE CRNT_TOTAL + TEMP6TOTAL TO TEMP6TOTAL
      @ LINE, 56 SAY STR(LAST_TOTAL,4)
      @ LINE, 65 SAY STR(CRNT_TOTAL,4)
      IF LAST_TOTAL = 0
        STORE CRNT_TOTAL * 100 TO PERCNTCHG
      ELSE
       STORE ROUND((((CRNT_TOTAL-LAST_TOTAL)/(LAST_TOTAL))*100),2) ;
        TO PERCNTCHG
      ENDIF
      @ LINE, 73 SAY STR(PERCNTCHG,4)
      STORE PLACE + 1 TO PLACE
      STORE LINE + 2 TO LINE
   ENDDO WHILE PLACE < 21
  STORE LINE + 1 TO LINE
  @ LINE, 1 SAY "TOTAL"
  @ LINE, 20 SAY STR(TEMP1TOTAL,4)
  @ LINE, 29 SAY STR(TEMP2TOTAL,4)
  @ LINE, 38 SAY STR(TEMP3TOTAL,4)
  @ LINE, 47 SAY STR(TEMP4TOTAL,4)
  @ LINE, 56 SAY STR(TEMP5TOTAL,4)
  @ LINE, 65 SAY STR(TEMP6TOTAL,4)
  IF TEMP5TOTAL = 0
    STORE TEMP6TOTAL * 100 TO PERCENTCHG
  ELSE
    STORE ROUND((((TEMP6TOTAL-TEMP5TOTAL)/TEMP5TOTAL)*100),2) ;
    TO PERCENTCHG
  ENDIF
  @ LINE, 73 SAY STR(PERCENTCHG,4)
  STORE COUNT + 1 TO COUNT
  RELEASE ALL LIKE TEMP*TOTAL
ENDDO WHILE COUNT < 3
EJECT
SET DEVICE TO SCREEN
RELEASE ALL EXCEPT USER*
RETURN
```

```
*******************************************************************************
* Program :      THIRD PRINT MODULE (PRINT3.PRG)                              *
* Author :       P. E. PAQUETTE                                               *
* Date :         1/2/88                                                       *
* Purpose :      THIS MODULE WILL FORMAT THE CRIME INFORMATION                 *
*                INTO THE FOUR TABULAR FORMATS REQUIRED BY THE                *
*                ANNUAL REPORT.                                               *
* Input File :   ARCHIVE.DBF                                                  *
* Output File :  NONE                                                         *
* Called By :    PRINT.PRG                                                    *
* Calls :        NONE                                                         *
* Variables                                                                   *
*   Local :      CENTER, HEADING, COUNTER, YR_TOTAL, LYR_TOAL                 *
*                T_HOMICIDE, T_RAPE, T_ROBBERY,T_ASSAULT                      *
*                T_BURGLARY, T_LARCENY,T_AUTO,CURNT_POP                       *
*                L_HOMICIDE, L_RAPE, L_ROBBERY, L_ASSAULT                     *
*                L_BURGLARY, L_LARCENY, L_AUTO, LAST_POP                      *
*                T_NARC, T_SELL, T_SMOKE, T_DEAL, L_SELL                      *
*                L_SMOKE, L_DEAL, PRCNTCHG, CT_RATE, LST_RATE                 *
*                TPRCNTCHG, C_TOTAL, L_TOTAL, PERCNTCHG, C_RATE               *
*                L_RATE, CRIME1, CRIME2, L_NARC                               *
*   Global :     NONE                                                         *
*******************************************************************************

@ 10, 3 CLEAR TO 20, 78
STORE 1 TO COUNTER
STORE 0 TO YR_TOTAL,LYR_TOTAL
SUM HOMICIDES,FORCERAPE,ROBBERY,ASSAULT,BURGLARY; FOR YEAR = CURNT_YEAR
TO T_HOMICIDE,T_RAPE,T_ROBBERY,T_ASSAULT,T_BURGLARY
SUM LARCENY,AUTOTHEFT,POPULATION;
 FOR YEAR = CURNT_YEAR TO T_LARCENY,T_AUTO,CURNT_POP

SUM HOMICIDES,FORCERAPE,ROBBERY,ASSAULT,BURGLARY;
 FOR YEAR = LAST_YEAR TO L_HOMICIDE,L_RAPE,L_ROBBERY,L_ASSAULT,L_BURGLARY
SUM LARCENY,AUTOTHEFT,POPULATION;
 FOR YEAR = LAST_YEAR TO L_LARCENY,L_AUTO,LAST_POP

SUM NARCCOUNT,SELLCOUNT,SMOKECOUNT,DEALCOUNT;
   FOR YEAR = CURNT_YEAR TO T_NARC,T_SELL,T_SMOKE,T_DEAL
SUM NARCCOUNT,SELLCOUNT,SMOKECOUNT,DEALCOUNT;
   FOR YEAR = LAST_YEAR TO L_NARC,L_SELL,L_SMOKE,L_DEAL

DO WHILE COUNTER < 12
  DO CASE
    CASE COUNTER = 1
      STORE T_HOMICIDE TO CRIME1
      STORE L_HOMICIDE TO CRIME2
       STORE "PRCNTCHG1" TO PRCNTCHG
      STORE "CT_RATE1" TO CT_RATE
      STORE "LST_RATE1"  TO LST_RATE
    CASE COUNTER = 2
      STORE T_RAPE TO CRIME1
      STORE L_RAPE TO CRIME2
      STORE "PRCNTCHG2" TO PRCNTCHG
      STORE "CT_RATE2" TO CT_RATE
```

```
   STORE "LST_RATE2"  TO LST_RATE
CASE COUNTER = 3
   STORE T_ROBBERY TO CRIME1
   STORE L_ROBBERY TO CRIME2
    STORE "PRCNTCHG3" TO PRCNTCHG
    STORE "CT_RATE3" TO CT_RATE
    STORE "LST_RATE3"  TO LST_RATE
CASE COUNTER = 4
    STORE T_ASSAULT TO CRIME1
    STORE L_ASSAULT TO CRIME2
    STORE "PRCNTCHG4" TO PRCNTCHG
    STORE "CT_RATE4" TO CT_RATE
    STORE "LST_RATE4"  TO LST_RATE
CASE COUNTER = 5
    STORE T_BURGLARY TO CRIME1
    STORE L_BURGLARY TO CRIME2
    STORE "PRCNTCHG5" TO PRCNTCHG
    STORE "CT_RATE5" TO CT_RATE
    STORE "LST_RATE5"  TO LST_RATE
CASE COUNTER = 6
    STORE T_LARCENY TO CRIME1
    STORE L_LARCENY TO CRIME2
    STORE "PRCNTCHG6" TO PRCNTCHG
    STORE "CT_RATE6" TO CT_RATE
    STORE "LST_RATE6"  TO LST_RATE
CASE COUNTER = 7
    STORE T_AUTO TO CRIME1
    STORE L_AUTO TO CRIME2
    STORE "PRCNTCHG7" TO PRCNTCHG
    STORE "CT_RATE7" TO CT_RATE
    STORE "LST_RATE7"  TO LST_RATE
CASE COUNTER = 8
    STORE T_DEAL TO CRIME1
    STORE L_DEAL TO CRIME2
    STORE "PRCNTCHG8" TO PRCNTCHG
    STORE "CT_RATE8" TO CT_RATE
    STORE "LST_RATE8"  TO LST_RATE
CASE COUNTER = 9
    STORE T_SMOKE TO CRIME1
    STORE L_SMOKE TO CRIME2
    STORE "PRCNTCHG9" TO PRCNTCHG
    STORE "CT_RATE9" TO CT_RATE
    STORE "LST_RATE9"  TO LST_RATE
CASE COUNTER = 10
    STORE T_SELL TO CRIME1
    STORE L_SELL TO CRIME2
    STORE "PRCNTCHG10" TO PRCNTCHG
    STORE "CT_RATE10" TO CT_RATE
    STORE "LST_RATE10"  TO LST_RATE
CASE COUNTER = 11
    STORE T_NARC TO CRIME1
    STORE L_NARC TO CRIME2
    STORE "PRCNTCHG11" TO PRCNTCHG
    STORE "CT_RATE11" TO CT_RATE
    STORE "LST_RATE11"  TO LST_RATE
```

```
  ENDCASE
  STORE ROUND((((CRIME1-CRIME2)/CRIME2)*100),2) TO &PRCNTCHG
  STORE ROUND(((1000*CRIME1)/CURNT_POP),4) TO &CT_RATE
  STORE ROUND(((1000*CRIME2)/LAST_POP),4) TO &LST_RATE
  IF COUNTER < 8
    STORE CRIME1 + YR_TOTAL TO  YR_TOTAL
    STORE CRIME2 + LYR_TOTAL TO LYR_TOTAL
   STORE ROUND((((YR_TOTAL-LYR_TOTAL)/LYR_TOTAL)*100),2) TO TPRCNTCHG
  ENDIF
  @ 12, 23 SAY "Summing totals, " + STR(11-COUNTER,2) + " TO GO"
  STORE COUNTER + 1 TO COUNTER
ENDDO WHILE COUNTER < 12

@ 0, 0
SET DEVICE TO PRINT
STORE "INDEX CRIMES REPORTED TO HQMC" TO HEADING
STORE (80 - LEN(TRIM("&HEADING")))/2 TO CENTER
@ 10, CENTER SAY "&HEADING"
@ 12, 33 SAY STR(LAST_YEAR,4) + "  VS  " + STR(CURNT_YEAR,4)
@ 16, 67 SAY "MOTOR"
@ 17, 16 SAY "FORCIBLE"
@ 17, 35 SAY "AGGREVATED"
@ 17, 66 SAY "VEHICLE"
@ 18, 0  SAY "YEAR"
@ 18, 6  SAY "HOMICIDE"
@ 18, 18 SAY "RAPE"
@ 18, 26 SAY "ROBBERY"
@ 18, 36 SAY "ASSAULTS"
@ 18, 47 SAY "BURGLARY"
@ 18, 57 SAY "LARCENY"
@ 18, 67 SAY "THEFT"
@ 18, 75 SAY "TOTAL"



@ 20, 1  SAY "CY"
@ 20, 10 SAY STR(L_HOMICIDE,4)
@ 20, 18 SAY STR(L_RAPE,4)
@ 20, 29 SAY STR(L_ROBBERY,4)
@ 20, 40 SAY STR(L_ASSAULT,4)
@ 20, 50 SAY STR(L_BURGLARY,4)
@ 20, 59 SAY STR(L_LARCENY,4)
@ 20, 69 SAY STR(L_AUTO,4)
@ 20, 75 SAY STR(LYR_TOTAL,4)
@ 21, 0  SAY STR(LAST_YEAR,4)

@ 23, 1 SAY "CY"
@ 23, 10 SAY STR(T_HOMICIDE,4)
@ 23, 18 SAY STR(T_RAPE,4)
@ 23, 29 SAY STR(T_ROBBERY,4)
@ 23, 40 SAY STR(T_ASSAULT,4)
@ 23, 50 SAY STR(T_BURGLARY,4)
@ 23, 59 SAY STR(T_LARCENY,4)
@ 23, 69 SAY STR(T_AUTO,4)
@ 23, 75 SAY STR(YR_TOTAL,4)
```

```
@ 24, 0  SAY STR(CURNT_YEAR,4)

@ 26, 2  SAY "%"
@ 27, 2  SAY "OF"
@ 27, 10 SAY STR(PRCNTCHG1,4)
@ 27, 18 SAY STR(PRCNTCHG2,4)
@ 27, 29 SAY STR(PRCNTCHG3,4)
@ 27, 40 SAY STR(PRCNTCHG4,4)
@ 27, 50 SAY STR(PRCNTCHG5,4)
@ 27, 59 SAY STR(PRCNTCHG6,4)
@ 27, 69 SAY STR(PRCNTCHG7,4)
@ 27, 75 SAY STR(TPRCNTCHG,4)
@ 28, 0  SAY "CHANGE"

@ 32, 2  SAY STR(LAST_YEAR,4)
@ 33, 2  SAY "RATE"
@ 33, 10 SAY STR(LST_RATE1,5,3)
@ 33, 18 SAY STR(LST_RATE2,5,3)
@ 33, 29 SAY STR(LST_RATE3,5,3)
@ 33, 40 SAY STR(LST_RATE4,5,3)
@ 33, 50 SAY STR(LST_RATE5,5,3)
@ 33, 59 SAY STR(LST_RATE6,5,3)
@ 33, 69 SAY STR(LST_RATE7,5,4)
@ 34, 0  SAY "PER 1000"

@ 36, 2  SAY STR(CURNT_YEAR,4)
@ 37, 2  SAY "RATE"
@ 37, 10 SAY STR(CT_RATE1,5,3)
@ 37, 18 SAY STR(CT_RATE2,5,3)
@ 37, 29 SAY STR(CT_RATE3,5,3)
@ 37, 40 SAY STR(CT_RATE4,5,3)
@ 37, 50 SAY STR(CT_RATE5,5,3)
@ 37, 59 SAY STR(CT_RATE6,5,3)
@ 37, 69 SAY STR(CT_RATE7,5,3)
@ 38, 0  SAY "PER 1000"
@ 41, 14 SAY "HOMICIDE INCLUDES MURDER AND NONNEGLIGENT MANSLAUGHTER"
EJECT

STORE T_HOMICIDE + T_RAPE + T_ROBBERY + T_ASSAULT   TO C_TOTAL
STORE L_HOMICIDE + L_RAPE + L_ROBBERY + L_ASSAULT   TO L_TOTAL
STORE ROUND((((C_TOTAL - L_TOTAL)/L_TOTAL)*100),2) TO PERCENTCHG
STORE ROUND(((1000*C_TOTAL)/CURNT_POP),4) TO C_RATE
STORE ROUND(((1000*L_TOTAL)/LAST_POP),4) TO L_RATE

@ 0,0
STORE "VIOLENT CRIMES REPORTED TO HQMC"  TO HEADING
STORE (80-LEN(TRIM("&HEADING")))/2 TO CENTER
@ 10, CENTER SAY "&HEADING"
@ 12, 33 SAY STR(LAST_YEAR,4) +"  VS  " + STR(CURNT_YEAR,4)

@ 17, 30 SAY "FORCIBLE"
@ 17, 53 SAY "AGGREVATED"
@ 18, 10 SAY "YEAR"
@ 18, 18 SAY "HOMICIDE"
@ 18, 32 SAY "RAPE"
```

```
@ 18, 42 SAY "ROBBERY"
@ 18, 54 SAY "ASSAULTS"
@ 18, 67 SAY "TOTAL"



@ 20, 11 SAY "CY"
@ 20, 20 SAY STR(L_HOMICIDE,4)
@ 20, 32 SAY STR(L_RAPE,4)
@ 20, 43 SAY STR(L_ROBBERY,4)
@ 20, 56 SAY STR(L_ASSAULT,4)
@ 20, 68 SAY STR(L_TOTAL,4)
@ 21, 10 SAY STR(LAST_YEAR,4)

@ 23, 11 SAY "CY"
@ 23, 20 SAY STR(T_HOMICIDE,4)
@ 23, 32 SAY STR(T_RAPE,4)
@ 23, 43 SAY STR(T_ROBBERY,4)
@ 23, 56 SAY STR(T_ASSAULT,4)
@ 23, 68 SAY STR(C_TOTAL,4)
@ 24, 10 SAY STR(CURNT_YEAR,4)

@ 26, 11 SAY "%"
@ 27, 11 SAY "OF"
@ 27, 20 SAY STR(PRCNTCHG1,4)
@ 27, 32 SAY STR(PRCNTCHG2,4)
@ 27, 43 SAY STR(PRCNTCHG3,4)
@ 27, 56 SAY STR(PRCNTCHG4,4)
@ 27, 68 SAY STR(PERCENTCHG,4)
@ 28, 9  SAY "CHANGE"

@ 32, 10 SAY STR(LAST_YEAR,4)
@ 33, 10 SAY "RATE"
@ 33, 17 SAY STR(LST_RATE1,5,3)
@ 33, 31 SAY STR(LST_RATE2,5,3)
@ 33, 42 SAY STR(LST_RATE3,5,3)
@ 33, 55 SAY STR(LST_RATE4,5,3)
@ 33, 67 SAY STR(L_RATE,5,3)
@ 34, 8  SAY "PER 1000"

@ 36, 10 SAY STR(CURNT_YEAR,4)
@ 37, 10 SAY "RATE"
@ 37, 17 SAY STR(CT_RATE1,5,3)
@ 37, 31 SAY STR(CT_RATE2,5,3)
@ 37, 42 SAY STR(CT_RATE3,5,3)
@ 37, 55 SAY STR(CT_RATE4,5,3)
@ 37, 67 SAY STR(C_RATE,5,3)
@ 38, 8  SAY "PER 1000"

@ 41, 14 SAY "HOMICIDE INCLUDES MURDER AND NONNEGLIGENT MANSLAUGHTER"

RELEASE C_TOTAL,L_TOTAL,PERCENTCHG,C_RATE,L_RATE
EJECT

STORE T_BURGLARY + T_LARCENY + T_AUTO TO C_TOTAL
```

```
STORE L_BURGLARY + L_LARCENY + L_AUTO TO L_TOTAL
STORE ROUND((((C_TOTAL - L_TOTAL)/L_TOTAL)*100),2) TO PERCENTCHG
STORE ROUND(((1000*C_TOTAL)/CURNT_POP),4) TO C_RATE
STORE ROUND(((1000*L_TOTAL)/LAST_POP),4) TO L_RATE

@ 0,0
STORE "CRIMES AGAINST PROPERTY REPORTED TO HQMC" TO HEADING
STORE (80-LEN(TRIM("&HEADING")))/2 TO CENTER
@ 10, CENTER SAY "&HEADING"
@ 12, 33 SAY STR(LAST_YEAR,4) + "  VS  " + STR(CURNT_YEAR,4)
@ 16, 55 SAY "MOTOR"
@ 17, 54 SAY "VEHICLE"
@ 18, 6  SAY "YEAR"
@ 18, 19 SAY "BURGLARY"
@ 18, 37 SAY "LARCENY"
@ 18, 55 SAY "THEFT"
@ 18, 71 SAY "TOTAL"



@ 20, 7  SAY "CY"
@ 20, 22 SAY STR(L_BURGLARY,4)
@ 20, 39 SAY STR(L_LARCENY,4)
@ 20, 55 SAY STR(L_AUTO,4)
@ 20, 72 SAY STR(L_TOTAL,4)
@ 21, 6  SAY STR(LAST_YEAR,4)

@ 23, 7 SAY "CY"
@ 23, 22 SAY STR(T_BURGLARY,4)
@ 23, 39 SAY STR(T_LARCENY,4)
@ 23, 55 SAY STR(T_AUTO,4)
@ 23, 72 SAY STR(C_TOTAL,4)
@ 24, 6  SAY STR(CURNT_YEAR,4)

@ 26, 7  SAY "%"
@ 27, 7  SAY "OF"
@ 27, 22 SAY STR(PRCNTCHG5,4)
@ 27, 39 SAY STR(PRCNTCHG6,4)
@ 27, 55 SAY STR(PRCNTCHG7,4)
@ 27, 72 SAY STR(PERCENTCHG,5,3)
@ 28, 5  SAY "CHANGE"

@ 32, 6  SAY STR(LAST_YEAR,4)
@ 33, 6  SAY "RATE"
@ 33, 21 SAY STR(LST_RATE5,5,3)
@ 33, 38 SAY STR(LST_RATE6,5,3)
@ 33, 54 SAY STR(LST_RATE7,5,4)
@ 33, 71 SAY STR(L_RATE,5,3)
@ 34, 4  SAY "PER 1000"

@ 36, 6  SAY STR(CURNT_YEAR,4)
@ 37, 6  SAY "RATE"
@ 37, 21 SAY STR(CT_RATE5,5,3)
@ 37, 38 SAY STR(CT_RATE6,5,3)
@ 37, 54 SAY STR(CT_RATE7,5,3)
```

```
@ 37, 71 SAY STR(C_RATE,5,3)
@ 38, 4  SAY "PER 1000"
EJECT

STORE T_NARC + T_SELL + T_SMOKE + T_DEAL TO C_TOTAL
STORE L_NARC + L_SELL + L_SMOKE + L_DEAL TO L_TOTAL
STORE ROUND((((C_TOTAL - L_TOTAL)/L_TOTAL)*100),2) TO PERCENTCHG
STORE ROUND(((1000*C_TOTAL)/CURNT_POP),4) TO C_RATE
STORE ROUND(((1000*L_TOTAL)/LAST_POP),4) TO L_RATE

@ 0, 0
STORE "NARCOTICS/DANGEROUS DRUGS AND MARIJUANA OFFENSES REPORTED " + ;
      "TO HQMC" TO HEADING
STORE (80-LEN(TRIM("&HEADING")))/2 TO CENTER
@ 10, CENTER SAY "&HEADING"
@ 12, 33 SAY STR (LAST_YEAR,4) + "  VS  " + STR(CURNT_YEAR,4)
SET MARGIN TO 5
@ 16, 38 SAY "NARCOTICS & DANGEROUS"
@ 17, 16 SAY "MARIJUANA"
@ 17, 46 SAY "DRUGS"
@ 18, 2  SAY " YEAR"
@ 18, 12 SAY "SALES"
@ 18, 24 SAY "POSSESSION"
@ 18, 42 SAY "SALES"
@ 18, 50 SAY "P0SSESSION"
@ 18, 64 SAY "TOTAL"

@ 20, 3  SAY "CY"
@ 20, 12 SAY STR(L_DEAL,4)
@ 20, 27 SAY STR(L_SMOKE,4)
@ 20, 40 SAY STR(L_SELL,4)
@ 20, 52 SAY STR(L_NARC,4)
@ 20, 64 SAY STR(L_TOTAL,4)
@ 21, 2  SAY STR(LAST_YEAR,4)

@ 23, 3  SAY "CY"
@ 23, 12 SAY STR(T_DEAL,4)
@ 23, 27 SAY STR(T_SMOKE,4)
@ 23, 40 SAY STR(T_SELL,4)
@ 23, 52 SAY STR(T_NARC,4)
@ 23, 64 SAY STR(C_TOTAL,4)
@ 24, 2  SAY STR(CURNT_YEAR,4)

@ 26, 3  SAY "%"
@ 27, 3  SAY "OF"
@ 27, 12 SAY STR(PRCNTCHG8,4)
@ 27, 27 SAY STR(PRCNTCHG9,4)
@ 27, 40 SAY STR(PRCNTCHG10,4)
@ 27, 52 SAY STR(PRCNTCHG11,4)
@ 27, 64 SAY STR(PERCENTCHG,4)
@ 28, 1  SAY "CHANGE"

@ 32, 2  SAY STR(LAST_YEAR,4)
@ 33, 2  SAY "RATE"
@ 33, 11 SAY STR(LST_RATE8,5,3)
```

```
@ 33, 26 SAY STR(LST_RATE9,5,3)
@ 33, 39 SAY STR(LST_RATE10,5,3)
@ 33, 51 SAY STR(LST_RATE11,5,3)
@ 33, 63 SAY STR(L_RATE,5,3)
@ 34, 0  SAY "PER 1000"

@ 36, 2  SAY STR(CURNT_YEAR,4)
@ 37, 2  SAY "RATE"
@ 37, 11 SAY STR(CT_RATE8,5,3)
@ 37, 26 SAY STR(CT_RATE9,5,3)
@ 37, 39 SAY STR(CT_RATE10,5,3)
@ 37, 51 SAY STR(CT_RATE11,5,3)
@ 37, 63 SAY STR(C_RATE,5,3)
@ 38, 0  SAY "PER 1000"
SET MARGIN TO 0
EJECT
SET DEVICE TO SCREEN
RELEASE ALL EXCEPT USER*
RETURN
```

```
*********************************************************************************
* Program :      FOURTH PRINT MODULE (PRINT4.PRG)                               *
* Author :       P. E. PAQUETTE                                                 *
* Date :         1/9/88                                                         *
* Purpose :      THIS MODULE WILL PRINT ALL THE CRIME                           *
*                INFORMATION IN THE LINE GRAPH FORMAT AS WELL                   *
*                AS CALCULATE THE CRIME RATE PER 1000 INDIVIDUALS               *
*                FOR EACH INSTALLATION.  IT WILL ALSO PROVIDE                   *
*                THE SAME INFORMATION FOR THE PRECEEDING YEAR AS                *
*                A MEANS OF VIEWING TRENDS                                      *
* Input File :   ARCHIVE.DBF                                                    *
* Output File :  NONE                                                           *
* Called By :    PRINT.PRG                                                      *
* Calls :        NONE                                                           *
* Variables                                                                     *
*   Local :      HEADING, COUNT, PLACE, CRIME, CENTER, LINE,                    *
*                STATION, START, COLMN,                                         *
*                L_RATE, C_RATE, DECREMENT                                      *
*   Global :     NONE                                                           *
*********************************************************************************

@ 10, 3  CLEAR TO 23,78
@ 12, 23 SAY "Starting last print sequence....."
STORE SPACE(62) TO HEADING
STORE 1 TO COUNT,PLACE
SET DEVICE TO PRINT
@ 0, 0
DO WHILE COUNT < 5
   DO CASE
    CASE COUNT =1
        STORE "INDEX CRIME RATE BY BASE (PER 1000)" TO HEADING
        STORE "INDEXTOTAL" TO CRIME
    CASE COUNT =2
        STORE "PROPERTY CRIME RATE BY BASE (PER 1000)" TO HEADING
         STORE "PROPCRIME" TO CRIME
    CASE COUNT =3
        STORE "DRUG CRIME RATE BY BASE (PER 1000)" TO HEADING
        STORE "DRUGTOTAL" TO CRIME
    CASE COUNT =4
        STORE "VIOLENT CRIME RATE BY BASE (PER 1000)" TO HEADING
         STORE "VIOLENT" TO CRIME
   ENDCASE
  STORE (80-LEN(TRIM("&HEADING")))/2 TO CENTER
  @ 5, CENTER SAY "&HEADING"
  STORE 10 TO LINE
  STORE 1 TO PLACE
  DO WHILE PLACE < 21
     DO CASE
       CASE PLACE =1
        STORE "MCLB ALBANY" TO STATION
       CASE PLACE =2
        STORE "MCLB BARSTOW" TO STATION
       CASE PLACE =3
        STORE "MCAS BEAUFORT" TO STATION
       CASE PLACE =4
```

```
    STORE "CAMP BUTLER" TO STATION
   CASE PLACE =5
    STORE "CHERRY POINT" TO STATION
   CASE PLACE =6
    STORE "MCAS EL TORO" TO STATION
   CASE PLACE =7
    STORE "CAMP ELMORE" TO STATION
   CASE PLACE =8
    STORE "HENDERSON HALL" TO STATION
   CASE PLACE =9
    STORE "MCAS IWAKUNI" TO STATION
   CASE PLACE =10
    STORE "MCAS KANEOHE" TO STATION
   CASE PLACE =11
    STORE "MCB CAMP LEJEUNE" TO STATION
   CASE PLACE =12
    STORE "MCRD PARRIS ISLAND" TO STATION
   CASE PLACE =13
    STORE "MCB CAMP PENDLETON" TO STATION
   CASE PLACE =14
    STORE "MCDEC QUANTICO" TO STATION
   CASE PLACE =15
    STORE "MCRD SAN DIEGO" TO STATION
   CASE PLACE =16
    STORE "MCAS TUSTIN" TO STATION
   CASE PLACE =17
    STORE "MCAGCC TWENTYNINE PALMS" TO STATION
   CASE PLACE =18
    STORE "MCAS YUMA" TO STATION
   CASE PLACE =19
    STORE "CAMP SMITH" TO STATION
   CASE PLACE =20
    STORE "MCAS NEW RIVER" TO STATION
ENDCASE
GO TOP
SET EXACT OFF
LOCATE FOR (STATION =  "&STATION") .AND. (YEAR = LAST_YEAR)
IF POPULATION = 0
  @ LINE, 30 SAY "NO POPULATION TOTAL LISTED FOR " + STR(LAST_YEAR,4)
ELSE
   STORE 32 TO START
   STORE 30 TO COLMN
  STORE ROUND(((1000*&CRIME)/POPULATION),3) TO L_RATE
   STORE (COLMN + L_RATE) TO DECREMENT
   DO WHILE START < DECREMENT
     @ LINE, COLMN SAY "*"
     STORE COLMN + 1 TO COLMN
     STORE (DECREMENT - 2) TO DECREMENT
  ENDDO WHILE START < DECREMENT
  @ LINE, (COLMN + 4) SAY STR(L_RATE,5,3)
ENDIF POPULATION
STORE LINE + 1 TO LINE
@ LINE, 5  SAY "&STATION"
GO TOP
LOCATE FOR (STATION = "&STATION") .AND. (YEAR = CURNT_YEAR)
```

```
    IF POPULATION = 0
      @ LINE, 30 SAY "NO POPULATION TOTAL LISTED FOR " + STR(CURNT_YEAR,4)
    ELSE
       STORE 32 TO START
       STORE 30 TO COLMN
       STORE ROUND(((1000*&CRIME)/POPULATION),3) TO C_RATE
       STORE (COLMN + C_RATE) TO DECREMENT
       DO WHILE START < DECREMENT
         @ LINE, COLMN SAY "="
         STORE COLMN + 1 TO COLMN
         STORE (DECREMENT - 2) TO DECREMENT
       ENDDO WHILE START < DECREMENT
       @ LINE, (COLMN + 4)  SAY STR(C_RATE,5,3)
    ENDIF POPULATION
    STORE LINE + 1 TO LINE
    STORE PLACE + 1 TO PLACE
  ENDDO WHILE PLACE < 21
 STORE COUNT + 1 TO COUNT
 STORE LINE + 2 TO LINE
 IF COUNT > 3
   @ LINE ,34 SAY "1"
   @ LINE ,39 SAY "2"
   @ LINE ,44 SAY "3"
   @ LINE ,49 SAY "4"
   @ LINE ,54 SAY "5"
   @ LINE ,59 SAY "6"
   @ LINE ,64 SAY "7"
   @ LINE ,69 SAY "8"
   @ LINE ,75 SAY "9"
 ELSE
   @ LINE ,34 SAY "10"
   @ LINE ,39 SAY "20"
   @ LINE ,44 SAY "30"
   @ LINE ,49 SAY "40"
   @ LINE ,54 SAY "50"
   @ LINE ,59 SAY "60"
   @ LINE ,64 SAY "70"
   @ LINE ,69 SAY "80"
   @ LINE ,75 SAY "90"
 ENDIF COUNT
 @ 54, 5 SAY "********** " + STR(LAST_YEAR,4)
 @ 55, 5 SAY "========== " + STR(CURNT_YEAR,4)
ENDDO WHILE COUNT < 5
SET SAFETY ON
EJECT
SET DEVICE TO SCREEN
RELEASE ALL EXCEPT USER*
RETURN
```

```
*********************************************************************************
* Program :      REPORTS.PRG                                                    *
* Author :       P. E. PAQUETTE                                                 *
* Date :         12/16/86                                                       *
* Purpose :      ALLOWS THE USER TO CHOOSE THE TYPE OF REPORT                   *
*                THAT IS GOING TO BE GENERATED.                                 *
* Input File :   NONE                                                           *
* Output File :  NONE                                                           *
* Called By :    MAIN.PRG                                                       *
* Calls :        MONTHLY.PRG, INFOANNUAL.PRG, PRINT, AUDIT.PRG                  *
* Variables                                                                     *
*   Local :      CHOICE, DELAY                                                  *
*   Global ;     NONE                                                           *
*********************************************************************************

STORE 9 TO CHOICE
DO WHILE CHOICE > 5
  CLEAR
  @ 0,0 TO 24,79 DOUBLE
  @ 3,31 SAY "MARINE CORPS"
  @ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 6,29 SAY "REPORTS GENERATION"
  @ 8,0  SAY CHR(204)
  @ 8,1 TO 8,78 DOUBLE
  @ 8,79 SAY CHR(185)
  @ 10,29 SAY "1.  Monthly Report"
  @ 12,29 SAY "2.  Compile Annual Report"
  @ 14,29 SAY "3.  Print Annual Report"
  @ 16,29 SAY "4.  Audit Report"
  @ 18,29 say "5.  Main Menu"
  @ 22,24 SAY "PLEASE ENTER A NUMBER"
  @ 22,46 GET CHOICE PICTURE "9"
  READ
  DO CASE
    CASE CHOICE = 1
      DO MONTHLY
    CASE CHOICE = 2
      DO INFOANNU
    CASE CHOICE = 3
      DO PRINT
    CASE CHOICE = 4
      DO AUDIT
    CASE CHOICE = 5
      CLEAR ALL
      RETURN
    OTHERWISE
      @ 10,17 CLEAR TO  19,55
      @ 10,17 TO 18,55 DOUBLE
      @ 14,25 SAY "ERROR IN INPUT VALUE"
      @ 15,25 SAY "ENTER A NUMBER LISTED"
      STORE 1 TO DELAY
      DO WHILE DELAY < 35
        STORE DELAY + 1 TO DELAY
      ENDDO WHILE DELAY < 35
  ENDCASE
```

```
ENDDO WHILE CHOICE > 5
RETURN
```

```
*********************************************************************************
* Program :      SECURITY.PRG                                                   *
* Author :       P. E. PAQUETTE                                                 *
* Date :         10/10/87                                                       *
* Purpose :      This program contains the password which allows                *
*                an authorized user to acess the Crime Statistics               *
*                Database. The user's name and rank as well as                  *
*                access information (files accessed, date & time                *
*                of acess) is recorded. This information is                     *
*                stored in a file called Access.dbf.                            *
* Input File :   NONE                                                           *
* Output File :  ACCESS.DBF                                                     *
* Called By :    N/A                                                            *
* Calls :        DISPLAY.PRG                                                    *
* Variables                                                                     *
*   Local :      RIGHT, LOOP, DELAY                                             *
*   Global :     USERNAME, USERRANK, USERUNIT,                                  *
*********************************************************************************

PUBLIC USERNAME, USERUNIT, USERRANK
STORE '                    ' TO USERNAME
STORE '    ' TO USERUNIT
STORE '    ' TO USERRANK
STORE 0 TO LOOP
STORE 'N' TO RIGHT
CLEAR
@ 0,0 TO 24,79 DOUBLE
@ 3,33 SAY "MARINE CORPS"
@ 4,22 SAY "CRIME STATISTICS REPORTING PROGRAM"
@ 5,22 SAY "    ACCESS TO THIS DATABASE IS "
@ 8,0 SAY CHR(204)
@ 8,79 SAY CHR(185)
@ 8,1 TO 8,78 DOUBLE
SET COLOR TO GR+/R*
@  7,34 SAY "RESTRICTED"
SET COLOR TO GR+/R,W/R,N
@  9,26 SAY "AUTHORIZED PERSONNEL  ONLY "
@ 14,26 SAY "What is your name and rank?"
@ 15,29 GET USERNAME
@ 17,25 SAY "What section do you work in?"
@ 18,37 GET USERUNIT
DO WHILE RIGHT = 'N'
 @ 20,25 SAY "Please type in the password:"
 SET COLOR TO R/R,W/R,N
 STORE 'XXXXXXXXXX' TO PASSWORD
 @ 21,34 GET password PICTURE '!!!!!!!!!!'
 READ
 SET COLOR TO GR+/R,W/R,N
 SET EXACT ON
 IF PASSWORD <>  'PAQMANXXXX'
   @ 10,19 CLEAR TO 19,57
   @ 10,19 TO 19,57 DOUBLE
   @ 14,27 SAY "PASSWORD INCORRECT !"
   @ 15,27 SAY "  TRY ONCE AGAIN "
   STORE LOOP + 1 TO LOOP
```

```
    ?? CHR(7)
  ELSE
    STORE 'Y' TO RIGHT
    RETURN
  ENDIF
  IF LOOP = 3
    @ 10,19 CLEAR TO 19,57
    @ 10,19 TO 19,57 DOUBLE
    @ 14,24 SAY "THREE GUESSES ARE ALL YOU GET"
    SET COLOR TO GR+/R*
    @ 15,30 SAY "  ACCESS DENIED"
    SET COLOR TO GR+/R,W/R,GR+
    ?? CHR(7)
    STORE 1 TO DELAY
        DO WHILE DELAY < 50
        STORE DELAY +1 TO DELAY
    ENDDO WHILE DELAY < 50
    CLEAR ALL
    QUIT
  ENDIF
ENDDO
RETURN
```

```
*****************************************************************************
* Program :      UPDATE.PRG                                                 *
* Author :       P. E. PAQUETTE                                             *
* Date :         10/10/87                                                   *
* Purpose :      THIS PROGRAM ALLOWS THE USER TO INPUT THE                  *
*                MONTHLY CRIME DATA FROM THE TWENTY REPORTING               *
*                INSTALLATIONS. DATA IS STORED IN A SEPARATE                *
*                DBF FILE FOR EACH INSTALLATION.                            *
* Input File :   ALL .DBF FILES                                             *
* Output File :  UPDATED .DBF FILES                                         *
* Called By :    MAIN.PRG                                                   *
* Calls :        NONE                                                       *
* Variables                                                                 *
*   Local :      CHOICE, STATION, EXIT, REPEAT, DBF, DELAY, DATE,           *
*                VERIFY, COUNT, CRIME, TF1, TF2, TF3, TF4, TF5,             *
*                TF6, TF7, TF8, TF9, TF10, TF11, TF12, TF13,                *
*                TF14  TF15, TF16, TF17, TF18, TF19, TF20, TF21,            *
*                TF22, TF23, TF24, TF25, TF26, TF27, TF28                   *
*   Global :     NONE                                                       *
*****************************************************************************

RELEASE ALL EXCEPT USER*
STORE 99 TO CHOICE
STORE "              " TO STATION
STORE "N" TO EXIT
STORE "Y" TO REPEAT
DO WHILE REPEAT = "Y"
DO WHILE EXIT = "N"
  DO WHILE choice > 20
  CLEAR
  @ 0,0 TO 24,79 DOUBLE
  @ 3,31 SAY "MARINE CORPS"
  @ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 6,32 SAY "UPDATE MENU"
  @ 8,0 SAY CHR(204)
  @ 8,79 SAY CHR(185)
  @ 8,1 TO 8,78 DOUBLE
  @ 10,15 SAY "1. MCLB ALBANY          11. MCB CAMP LEJEUNE"
  @ 11,15 SAY "2. MCLB BARSTOW          12. MCRD PARRIS ISLAND"
  @ 12,15 SAY "3. MCAS BEAUFORT         13. MCB CAMP PENDLETON"
  @ 13,15 SAY "4. CAMP BUTLER          14. MCDEC QUANTICO"
  @ 14,15 SAY "5. MCAS CHERRY POINT      15. MCRD SAN DIEGO"
  @ 15,15 SAY "6. MCAS EL TORO          16. MCAS TUSTIN"
  @ 16,15 SAY "7. CAMP ELMORE           17. MCAGCC 29 PALMS"
  @ 17,15 SAY "8. HENDERSON HALL         18. MCAS YUMA"
  @ 18,15 SAY "9. MCAS IWAKUNI          19. CAMP SMITH"
  @ 19,14 SAY "10. MCAS KANEOHE          20. MCAS NEW RIVER"
  @ 21,18 SAY "SELECT THE STATION YOU WISH TO UPDATE"
  @ 22,18 SAY "  ENTER '99' TO RETURN TO MAIN MENU"
  @ 23,37 GET CHOICE PICTURE "99"
  READ
  DO CASE
    CASE CHOICE =1
        STORE "MCLB ALBANY" TO STATION
        STORE "ALBANY" TO DBF
```

```
CASE CHOICE =2
    STORE "MCLB BARSTOW" TO STATION
    STORE "BARSTOW" TO DBF
CASE CHOICE =3
    STORE "MCAS BEAUFORT" TO STATION
    STORE "BEAUFORT" TO DBF
CASE CHOICE =4
    STORE "CAMP BUTLER" TO STATION
    STORE "BUTLER" TO DBF
CASE CHOICE =5
    STORE "CHERRY POINT" TO STATION
    STORE "CHERRY" TO DBF
CASE CHOICE =6
    STORE "MCAS EL TORO" TO STATION
    STORE "ELTORO" TO DBF
CASE CHOICE =7
    STORE "CAMP ELMORE" TO STATION
    STORE "ELMORE" TO DBF
CASE CHOICE =8
    STORE "HENDERSON HALL" TO STATION
    STORE "HENDRSON" TO DBF
CASE CHOICE =9
    STORE "MCAS IWAKUNI" TO STATION
    STORE "IWAKUNI" TO DBF
CASE CHOICE =10
    STORE "MCAS KANEOHE" TO STATION
    STORE "KANEOHE" TO DBF
CASE CHOICE =11
    STORE "MCB CAMP LEJEUNE" TO STATION
    STORE "LEJEUNE" TO DBF
CASE CHOICE =12
    STORE "MCRD PARRIS ISLAND" TO STATION
    STORE "PI" TO DBF
CASE CHOICE =13
    STORE "MCB CAMP PENDLETON" TO STATION
    STORE "PENDELTN" TO DBF
CASE CHOICE =14
    STORE "MCDEC QUANTICO" TO STATION
    STORE "QUANTICO" TO DBF
CASE CHOICE =15
    STORE "MCRD SAN DIEGO" TO STATION
    STORE "SANDIEGO" TO DBF
CASE CHOICE =16
    STORE "MCAS TUSTIN" TO STATION
    STORE "TUSTIN" TO DBF
CASE CHOICE =17
    STORE "MCAGCC TWENTYNINE PALMS" TO STATION
    STORE "STUMPS" TO DBF
CASE CHOICE =18
    STORE "MCAS YUMA" TO STATION
    STORE "YUMA" TO DBF
CASE CHOICE =19
    STORE "CAMP SMITH" TO STATION
    STORE "SMITH" TO DBF
CASE CHOICE =20
```

```
        STORE "MCAS NEW RIVER" TO STATION
        STORE "NEWRIVER" TO DBF
    CASE CHOICE = 99
        RETURN
    OTHERWISE
        @ 10,17 CLEAR TO 19,55
        @ 10,17 TO 18,55 DOUBLE
        @ 14,25 SAY "ERROR IN INPUT VALUE"
        @ 15,25 SAY "ENTER A NUMBER LISTED"
        ?? CHR(7)
        STORE 1 TO DELAY
        DO WHILE DELAY < 50
          STORE DELAY +1 TO DELAY
        ENDDO WHILE DELAY < 50
        CHOICE = 99
    ENDCASE
  ENDDO WHILE CHOICE > 20
  STORE  "MM/YY"   TO DATE
  STORE 23 TO MONTH
  DO WHILE (MONTH < 1) .OR. (MONTH > 12)
    @ 9,01 CLEAR TO 23,78
    @ 14,25 SAY "WHAT IS THE DATE OF THE"
    @ 15,25 SAY "  REPORT TO BE ADDED?"
    @ 16,34 GET DATE PICTURE "99/99"
    READ
    MONTH = VAL(SUBSTR(DATE,1,2))
    IF (MONTH < 1) .OR. (MONTH > 12)
      @ 9,1 CLEAR TO 23,78
      @ 12, 18 SAY "  SORRY ONLY TWELVE MONTHS IN A YEAR !"
      @ 14, 18 SAY "PICK A NUMBER BETWEEN 1 AND 12 INCLUSIVE"
      ?? CHR(7)
      STORE 1 TO DELAY
      DO WHILE DELAY < 50
        STORE DELAY +1 TO DELAY
      ENDDO WHILE DELAY < 50
    ENDIF
  ENDDO WHILE MONTH
  CLEAR
  @ 0,0 TO 24,79 DOUBLE
  @ 3,31 SAY "MARINE CORPS"
  @ 4,20 SAY "CRIME STATISTICS REPORTING PROGRAM"
  @ 6,28 SAY "VALIDATION OF INPUT"
  @ 8,0 SAY CHR(204)
  @ 8,79 SAY CHR(185)
  @ 8,1 TO 8,78 DOUBLE
  @ 10,7 SAY "THE INFORMATION WHICH YOU HAVE PROVIDED WILL BE USED TO
SEARCH"
  @ 11,7 SAY "THE CRIME REPORTING STATISTICS DATA BASE.  PLEASE CHECK THE"
  @ 12,7 SAY "INFORMATION BELOW TO ENSURE THAT YOUR ENTRIES ARE CORRECT."
  @ 16,26 SAY "INSTALLATION:  &STATION"
  @ 18,26 SAY "DATE OF REPORT:   &DATE"
  @ 22,26 SAY "ARE ALL ENTRIES CORRECT?"
  STORE "Y" TO VERIFY
  @ 23,37 GET VERIFY PICTURE "!"
  READ
```

```
  IF VERIFY = "N"
    STORE "N" TO EXIT
    STORE 99 TO CHOICE
  ELSE
    STORE "Y" TO EXIT
  ENDIF
ENDDO WHILE EXIT = N
CLEAR
@  0,  0  TO 24, 79    DOUBLE
@  5,  0  SAY CHR(204)
@  5, 79  SAY CHR(185)
@  5,  1  TO  5, 78  DOUBLE
@  2, 26  SAY "INSTALLATION: &STATION"
@  3, 26  SAY "DATE OF REPORT: &DATE"
USE &DBF
GO TOP
STORE 0 TO COUNT
STORE "                    " TO CRIME
DO WHILE COUNT < 28
  STORE COUNT + 1  TO COUNT
  DO CASE
    CASE COUNT = 1
       STORE "MURDER" TO CRIME
    CASE COUNT = 2
       STORE "MANSLAUGHTER" TO CRIME
    CASE COUNT = 3
       STORE "RAPE" TO CRIME
    CASE COUNT = 4
       STORE "ATTEMPTED RAPE" TO CRIME
    CASE COUNT = 5
       STORE "ROBBERY W/FIRE ARM" TO CRIME
    CASE COUNT = 6
       STORE "ROBBERY W/OTHER WEAPON" TO CRIME
    CASE COUNT = 7
       STORE "ASSAULT W /FIRE ARM" TO CRIME
    CASE COUNT = 8
       STORE "ASSAULT W/OTHER WEAPON" TO CRIME
    CASE COUNT = 9
       STORE "SIMPLE ASSAULT" TO CRIME
    CASE COUNT = 10
       STORE "ENTRY" TO CRIME
    CASE COUNT = 11
       STORE "ATTEMPTED ENTRY" TO CRIME
    CASE COUNT = 12
       STORE "GOVT. PROPERTY" TO CRIME
    CASE COUNT = 13
       STORE "NON-GOVT. PROPERTY" TO CRIME
    CASE COUNT = 14
       STORE "LARCENY OF DOD PROPERTY" TO CRIME
    CASE COUNT = 15
       STORE "LARCENY OF NON-DOD PROPERTY" TO CRIME
    CASE COUNT = 16
       STORE "AUTO THEFT" TO CRIME
    CASE COUNT = 17
       STORE "OTHER VEHICLE THEFT" TO CRIME
```

```
  CASE COUNT = 18
     STORE "THEFT OF GOVT. VEHICLE" TO CRIME
  CASE COUNT = 19
     STORE "THEFT OF NON-GOVT. VEHICLE" TO CRIME
  CASE COUNT = 20
      STORE "NARCOTICS USE & POSSESSION" TO CRIME
  CASE COUNT = 21
     STORE "NARCOTICS SALE & TRAFFICKING" TO CRIME
  CASE COUNT = 22
     STORE "MARIJUANA USE & POSSESSION" TO CRIME
  CASE COUNT = 23
     STORE "MARIJUANA SALE & TRAFFICKING" TO CRIME
  CASE COUNT = 24
     STORE "DWI ON BASE" TO CRIME
  CASE COUNT = 25
     STORE "DWI OFF BASE" TO CRIME
  CASE COUNT = 26
     STORE "DUI ON BASE" TO CRIME
  CASE COUNT = 27
     STORE "DUI OFF BASE" TO CRIME
ENDCASE
 STORE 00 TO TF1,TF2,TF3,TF4,TF5,TF6,TF7,TF8,TF9,TF10,TF11
 STORE 00 TO TF12,TF13,TF14,TF15,TF16,TF17,TF18,TF19,TF20
 STORE 00 TO TF21,TF22,TF23,TF24,TF25,TF26,TF27,TF28
 @  4, 20  CLEAR  TO 4, 75
 @  4, 20  SAY "CLASSIFICATION OF OFFENSE:  &CRIME"
 @  6,  2  SAY "NUMBER OF OFFENSES ON BASE:"
 @  6, 30  GET  TF1 picture '99'
 @  6, 37  SAY "OFF BASE:"
 @  6, 47  GET  TF2 picture '99'
 @  7,  2  SAY "NUMBER OF UNFOUNDED FALSE REPORTS ON BASE:"
 @  7, 45  GET  TF3 picture '99'
 @  7, 50  SAY "OFF BASE:"
 @  7, 60  GET  TF4 picture '99'
 @  8,  2  SAY "NUMBER CLEARED BY ARREST:"
 @  8, 28  GET  TF5 picture '99'
 @  9,  2  SAY "NUMBER INVESTIGATED BY NIS:"
 @  9, 30  GET  TF6 picture '99'
 @ 11, 32  SAY "PERPETRATOR       VICTIM"
 @ 12,  2  SAY "USMC:"
 @ 13,  2  SAY "OTHER SERVICES:"
 @ 14,  2  SAY "DEPENDANTS:"
 @ 15,  2  SAY "DOD CIVILIAN PERSONNEL:"
 @ 16,  2  SAY "MALES:"
 @ 17,  2  SAY "FEMALES:"
 @ 18,  2  SAY "CAUCASIAN:"
 @ 19,  2  SAY "NEGRO:"
 @ 20,  2  SAY "ALL OTHERS:"
 @ 12, 38 GET TF9  picture '99'
 @ 13, 38 GET TF10 picture '99'
 @ 14, 38 GET TF11 picture '99'
 @ 15, 38 GET TF12 picture '99'
 @ 16, 38 GET TF13 picture '99'
 @ 17, 38 GET TF14 picture '99'
 @ 18, 38 GET TF15 picture '99'
```

```
@ 19, 38 GET TF16 picture '99'
@ 20, 38 GET TF17 picture '99'
@ 12, 54 GET TF18 picture '99'
@ 13, 54 GET TF19 picture '99'
@ 14, 54 GET TF20 picture '99'
@ 15, 54 GET TF21 picture '99'
@ 16, 54 GET TF22 picture '99'
@ 17, 54 GET TF23 picture '99'
@ 18, 54 GET TF24 picture '99'
@ 19, 54 GET TF25 picture '99'
@ 20, 54 GET TF26 picture '99'
@ 21,  2  SAY "DRUG INVOLVEMENT:"
@ 21, 38  GET  TF27 picture '99'
@ 22,  2  SAY "ALCOHOL INVOLVEMENT:"
@ 22, 38  GET  TF28 picture '99'
READ
STORE "Y" TO VERIFY
@ 23,23 CLEAR TO 23,51
?? CHR(7)
@ 23, 24 SAY "ARE ALL ENTRIES CORRECT ?"
@ 23, 50 GET VERIFY PICTURE "!"
READ
IF VERIFY = "Y"
  STORE "Y" TO EXIT
  APPEND BLANK
  REPLACE F1 WITH TF1
  REPLACE F2 WITH TF2
  REPLACE F3 WITH TF3
  REPLACE F4 WITH TF4
  REPLACE F5 WITH TF5
  REPLACE F6 WITH TF6
  REPLACE F9 WITH TF9
  REPLACE F10 WITH TF10
  REPLACE F11 WITH TF11
  REPLACE F12 WITH TF12
  REPLACE F13 WITH TF13
  REPLACE F14 WITH TF14
  REPLACE F15 WITH TF15
  REPLACE F16 WITH TF16
  REPLACE F17 WITH TF17
  REPLACE F18 WITH TF18
  REPLACE F19 WITH TF19
  REPLACE F20 WITH TF20
  REPLACE F21 WITH TF21
  REPLACE F22 WITH TF22
  REPLACE F23 WITH TF23
  REPLACE F24 WITH TF24
  REPLACE F25 WITH TF25
  REPLACE F26 WITH TF26
  REPLACE F27 WITH TF27
  REPLACE F28 WITH TF28
  REPLACE DATE WITH "&DATE"
  REPLACE CRIME WITH "&CRIME"
  @ 23, 24 CLEAR TO 23,51
ELSE
```

```
    STORE COUNT - 1 TO COUNT
    STORE "N" TO EXIT
  ENDIF
ENDDO WHILE COUNT < 28
CLEAR
4, 2   CLEAR TO 5, 78
@  0,  0  TO 24, 79    DOUBLE
@  5,  0  SAY CHR(204)
@  5, 79  SAY CHR(185)
@  5,  1  TO  5, 78 DOUBLE
@ 12, 23 SAY "WOULD YOU LIKE TO UPDATE ANOTHER INSTALLATION ?"
@ 13, 39 GET ANS PICTURE "Y"
IF ANS = "Y"
  STORE "Y" TO REPEAT
  CLOSE DATABASES
ELSE
  CLOSE DATABASES
  RELEASE ALL EXCEPT USER*
ENDIF
ENDDO WHILE REPEAT = "Y"
RETURN
```

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	1
3.	Commandt of the Marine Corps Attn: GySgt Shorjes Headquarters Marine Corps POS-43 Washington, D. C. 20380-0001	1
4.	Department Chairman, Code 54 Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
5.	Computer Technology Curriculum Office, Code 37 Naval Postgraduate School Monterey, California 93943-5000	1
6.	Lieutenant Commander Barry A. Frew, Code 54 FW Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
7.	Professor Benjamin J. Roberts, Code 54 RO Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
8.	Captain Paul E. Paquette 3 Cone Street Winslow, Maine 04902	2

Thesis
P1478
c.1

Paquette
A proposal for a micro-computer based system to automate the Marine Corps Crime Statistics Report-ing Program.